Bibliography

[1] Michael R. Anderberg. *Cluster Analysis for Applications.* Academic Press, San Francisco, California, 1973.

[2] Richard O. Duda and Peter E. Hart. *Pattern Recognition.* John Wiley and Sons, New York, 1973.

[3] R. Jagannathan and Teresa Lunt. System design document: Next generation intrusion detection expert system (NIDES). SRI report, SRI International, Menlo Park, California, March 9, 1993.

[4] T. F. Lunt, Ann Tamaru, Fred Gilham, R. Jagannathan, Caveh Jalali, H. S. Javitz, A. Valdes, P. G. Neumann, and T. D. Garvey. A Real-Time Intrusion-Detection Expert System (IDES), Final Technical Report. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, February 1992.

[5] S. Karlin. *A First Course in Stochastic Processes.* Academic Press, New York, 1969.

[6] R.A. Olshen L. Breiman, J.H. Friedman and C.J. Stone. *Classification and Regression Trees.* Wadsworth and Brooks/Cole Advanced Books and Software, Monterey, California, 1984.

[7] *Wisdom and Sense Guidebook.* Los Alamos National Laboratory, Los Alamos, New Mexico. Undated.

[8] Donald F. Morrison. *Multivariate Statistical Methods.* McGraw-Hill, San Francisco, California, 1976.

[9] G. Liepins P. Helman and W. Richards. Foundations of intrusion detection. In *Proceedings of the IEEE Symposium on Computer Security*, 1992.

[10] G. Shafer and editors J. Pearl. Readings in uncertain reasoning. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1990.

[11] S. Smaha. Haystack audit trail analysis system. Status Report HS_STAT.TXT, Haystack Laboratories, Colorado, August 15, 1990.

We were pleasantly surprised to find that the NIDES statistical component performed as well as it did in the Safeguard environment, given its stark differences with the SRI Sun-Unix environment, and the relatively small changes in the algorithm parameters that were implemented. This gives us reason to be optimistic that the NIDES statistical component can be applied in other circumstances, which were not in our minds when the algorithm was developed. However, as mentioned earlier, there is no substitute for actually testing the algorithm in those environments.

tistical approaches because other approaches have not been available to us to test until relatively recently, nor have we been funded to perform such testing. We believe that such testing could be very beneficial to statistical algorithm development, not so much to determine which statistical algorithms are better, but rather to contribute to an understanding of why they perform well or not, so that we can use such knowledge to improve them further.

Because our most important sources of audit record data were our own Sun-Unix systems and because the immediate client's systems were also Sun-Unix, there are undoubtedly aspects of our statistical component that are rather "tailored" to these types of computer systems and users. We have also had an opportunity to test our system in two other very different environments. One of these environments was IBM mainframe audit record data gathered in field offices of a federal agency, where the predominant usage was among clerks entering and retrieving data from a fixed number of large databases. The second use was data where the subjects were applications, and the intent of using the NIDES statistical component was to differentiate among applications to detect masquerading by more resource intensive applications (constituting misuse of exported computers). In all three environments, the NIDES statistical system has performed very well. The last environment (which was known as "Safeguard" because the intent was to safeguard the usage of exported computers) is particularly noteworthy because it was amost diametrically opposed (in many important statistical ways) to the SRI Sun-Unix environment in which NIDES was developed:

- The subjects in the SRI Sun-Unix environment are sophisticated computer users, while the subjects in Safeguard are applications.

- The volume of audit records in the SRI Sun-Unix environment averages about 2,000 per user-day; the volume in the Safeguard environment averages only about 10 per day.

- Individual audit records in the SRI Sun-Unix environment contain only a small fragment of the information necessary to determine the behavior of a user; each individual audit record in the Safeguard environment contains complete information about an invocation of an application.

- There is no natural beginning and ending to an SRI Sun-Unix user's activity, whereas in the Safeguard environment each audit record represents a complete invocation of an application and is the natural analysis unit.

- In the SRI Sun-Unix environment the most distinguishing measures tend to be those which are categorical, and therefore the NIDES statistical approach was developed with categorical measures in mind; in the Safeguard environment all measures are counting measures.

most recent records have "triggered" or "touched" different measures. This provides an overview of the general types of activity that have recently taken place. For example, the audit record distribution measure would examine the percentage of recent audit records that concern a file access (that is, touch the file access measure). The audit record distribution measure uses exponential weighting based on counts of audit records. The remaining measures track the type of behavior that has transpired within a measure. For example, the file access measure examines which specific files were recently accessed, and whether or not these were unusual files.

In previous versions of NIDES, we did not subdivide the volume of activity from the specific behavior within that activity. This resulted in an inability to distinguish a relatively low volume of highly suspicious behavior from a high volume of behavior with typical levels of suspiciousness. The heuristic solution was to "normalize" our measures by the volume of activity; this was functionally equivalent to weighting on the basis of audit record counts and separately tracking audit record volume. This change to weighting on the basis of audit record count also facilitated the construction of short-term and long-term profiles, and the use of the chi-square statistic to compare those profiles. (Exponential weighting based on time is not compatible with the chi-square approach to profile comparison, and, as a result, the three intensity measures use a different approach.) There is a secondary advantage to the separation of audit record volume from the type of activity in those records — the measures are easier to interpret. We can separately detect whether the volume of activity is too high, whether the general type of activity within that volume is unusual (and in what ways it is unusual), and whether specific behavior within the activity is unusual (and in what ways it is unusual).

4.9 Q: For What Types of Computer Systems Will the NIDES Statistical Component be Most Applicable? Least Applicable?

The NIDES statistical component was developed to be very general and applicable across a wide range of systems and types of "subjects." Thus, it is our belief that the NIDES statistical component should perform well in many circumstances. However, there is no way of knowing exactly how well the statistical component will perform without actually testing it.

It is important to distinguish between relative and absolute performance. Some circumstances are generally favorable to statistical anomaly detection, including stable subject profiles, users with sharply defined and restricted behavior, and substantial differences in misuse and normal behavior. Statistical systems, including NIDES, will tend to perform better in these circumstances. Conversely, when these circumstances are missing, all statistical systems will tend to yield poor results. We have no way of knowing how well the NIDES statistical component performs relative to other sta-

4.7 Q: Why Does the NIDES Statistical Algorithm Treat Counting Measures as if They Were Categorical Measures?

The NIDES statistical algorithm converts counting measures to categorical measures by defining categories that are ranges of counts. We do so because we wanted to be very general with respect to the types of changes in counting distributions that we would detect as anomalous. For example, consider a user who has two modes of using the computer — reading mail and running large spreadsheets. The first type of activity would result in low CPU usage, and the second type of activity would result in high CPU usage. Therefore, this user's CPU usage would tend to be bimodal, with very few audit records recording intermediate CPU values. In such a circumstance, if we observed a large number of audit records that reflected activity with an average amount of CPU usage (that is, audit records in the rarely occurring center bins of the CPU usage distribution) we would tend to be suspicious. Such activity could not be detected if we only examined statistics such as mean CPU usage. Rather, it would be necessary to examine the entire distribution of CPU usage. We found that the most convenient method for examining the entire distribution for counting measures was to redefine them as categorical measures. This allowed us to use the already-existing methodology for categorical measures.

Other statistical methods for examining the entire distribution of counting measures have been developed in the statistical literature. For example, Kolmogorov statistics allow the comparison of an empirical cumulative distribution with a long-term cumulative distribution. However, such procedures tend to either be computationally intensive or require substantial amounts of storage, and few have been adapted

4.8 Q: Why Does the NIDES Statistical Algorithm Use Exponential Weighting Based on Counts of Audit Records (rather than Clock Time) for Nearly All of the Measures?

The NIDES statistical algorithm differentiates between the volume of audit records that are currently being received, the general type of activity that is represented in those audit records, and the specific type of behavior that is occurring within each measure. For the purpose of monitoring the volume of audit records we use three audit record intensity measures, which currently have half-lives of 1, 10, and 60 minutes. For these three measures the exponential weighting is based on time. A single measure, denoted audit record distribution, tracks what percentage of the

any other categorical measure. There is no need to limit this consideration to pairs; this approach would apply equally well for triplets, quadruplets, and so forth.

The theoretical advantage of considering pairs of measures (or equivalently, of constructing new measures that are two-tuples of existing measures) is that interactions between measures can be examined. Continuing our example, suppose that user A (who works 8 a.m. to 5 p.m. on weekdays) accesses file B daily, but only during the hour after lunch. One day, however, file B is used frequently between 9 a.m. and 10 a.m. The NIDES approach as currently implemented would not detect such usage as anomalous, because it is not unusual for user A to use file B, nor is it unusual for this user to access the computer during these hours. The use of a two-tuple approach would be necessary for NIDES to detect as anomalous the use of file B during the morning hours.

We have not added a mechanism in the NIDES computer program for constructing new measures from groups of other measures. The primary reason for not doing so is that we have concerns that the number of categories in composite measures would become so large, and the frequency with which each category is seen would be so small, that the NIDES methodology would cease to function well. The NIDES methodology works best when user behavior is concentrated in relatively few categories.

4.6 Q: Why Does the NIDES Statistical Algorithm Use Exponentially Weighted Sums?

The NIDES statistical algorithm uses exponentially weighted sums, rather than a window of past values, for three primary reasons. First, we want to reduce storage requirements and accelerate processing. Use of a window of past values would require that we store all the past values in the window. Each time a new audit record of the appropriate type arrived, we would have to recall those values, delete the oldest value, and add the newest value to the list. We would then recalculate our test statistic to accommodate the deletion of the oldest value and the addition of the newest value. This is considerably more effort and requires more storage than multiplying our summary weighted sum statistic by a decay constant and adding in the value of the newest audit record. Second, we use an exponentially weighted sum because we do not have any "natural" boundaries that delineate past from current behavior and therefore would not know how long the window should extend into the past. Third, we do want more distant behavior to count less towards anomalies than more recent behavior. This does not occur with a window approach (except that audit records further in the past than the window boundary do not count towards anomaly detection), but occurs smoothly and gradually with an exponentially weighted sum approach.

of probability distributions that are not like the normal probability distribution. Their principal deficiency is that they are not as powerful as statistical tests formulated on the basis of a priori knowledge of the specific probability distribution of misuse behavior. That is, the chi-square-type statistics spread their power over a broad (in fact, infinite) range of probability distributions of misuse behavior.

- The NIDES statistical component does not assume independence of measures under either the normal or misuse behavior, nor does its validity depend on the independence of measures (that is, its false positive and false negative rates are uninfluenced by the dependence of measures). However, because the NIDES algorithm as currently configured does not place great emphasis on the correlation among measures, it is quite compatible with the assumption of independence of misuse behavior. In combination with the comments earlier about the uniformity assumption, we would expect the NIDES statistical algorithm to perform well when the misuse behavior is specified to be uniform and independent, although it will not perform as well as the optimal statistical test.

- Another type of misuse behavior that has not been explicitly considered by Helman is when the misuse behavior does not affect the marginal distributions of any measure, but does affect the correlation among the measures. As currently configured, the NIDES statistical component will not be particularly powerful in detecting this type of misuse behavior (unless the direction of change is to increase the correlation coefficient among the measures). However, it is possible to modify the algorithm in such a way that correlations are explicitly considered, thus, considerably strengthening the ability to detect changes in correlations. We view this as a potentially valuable area of future research.

4.5 Q: Can the NIDES Statistical Algorithm Consider Pairs of Measures Simultaneously (for example, command name and file name used)?

Theoretically, the NIDES statistical component can consider pairs of measures simultaneously, although this requires defining a new measure that is the cross product of two other measures. For example, consider the measure of name of file accessed and the measure of time of day. A composite measure might be defined as the two-tuple consisting of the name of the file accessed and the time of day of the access (perhaps categorized by hour of day). Each value for the two-tuple is a category for the new composite measure. The composite measure could thus, be handled by the NIDES statistical measure as a categorical measure in essentially the same way as

not currently part of NIDES, the NIDES statistical algorithm could be rewritten so that any two measures that are currently considered separately could also be considered jointly. For example, command name and file name could be jointly considered as a single measure, which takes categorical measures that are specified by both the command name and the file name.)

4.4 Q: What is the Relationship of the NIDES Statistical Algorithm to the Work of Helman et al.?

As mentioned earlier, the approach of Helman et al. is essentially to derive an optimal test statistic under the assumption that the probability distribution for misuse behavior is known. Helman et al. realize, however, that in practice probability distributions for misuse behavior will probably not be known. They therefore discuss two assumptions that can be made that dramatically simplify forms for these probability distributions:

- Independence — Under this assumption each measure (denoted by Helman as an attribute) is independent of each other attribute.
- Uniformity — Under this assumption, misuse behavior for each measure is assumed to be uniformly distributed over the possible values for the measure.

Because the NIDES statistical component does not assume a known probability distribution for misuse behavior, it cannot be easily related to Helman's approach. But we can make some observations:

- We do not believe that assumptions of independence or uniformity of misuse behavior are justified. Optimal procedures for detecting independent and uniform misuse behavior might be very poor in detecting actual misuse behavior. Nevertheless, it is desirable, in the absence of knowledge about misuse behavior, that an anomaly detector be at least capable of detecting independent and uniform misuse behavior.
- Because the NIDES statistical component uses a chi-square-like statistic to measure the differences between the short-term and long-term probability distributions for each individual measure, it will be capable of (and probably fairly good at) detecting uniform and independent misuse behavior.
- The NIDES statistical component can detect misuse behavior that is neither uniform nor independent. This is particularly true with respect to nonuniformity. Chi-square-type statistics are good at detecting an extremely broad range

by processes executed by the user). In this event, the space X has 100,000 possible values (although only a fraction of these possible values might actually have been experienced). Add measures for host used, time of day, CPU time used, I/O amount, memory usage, and so forth, and the potential size of the space X becomes enormous. Conservatively, the size of the space X would contain trillions of possible values. Even if we limit the space X to those values actually experienced by the user, X would contain millions or billions of values. Simply enumerating these values (so we can keep a record of their historical non-zero probabilities) might overwhelm our storage capability, and processing this list would unacceptably slow down our algorithm.

In addition to processing capability problems inherent in considering the full space of outcomes, there is an even greater problem concerning the sample size requirements. As noted by Helman et al., "the sample size mandated by the enormity of the event space [X] ... might exceed memory limitations, or perhaps, the [normal and misuse] processes ..." cannot be assumed to be stationary over the time required to accumulate the requisite historical database." By the time we gather enough data to obtain reliable probabilities for the events in X, the user will have changed job assignments, changed jobs, or possibly died. Even if user behavior were extremely stable over time, we doubt that any system security officer would be willing to wait one or more years before starting anomaly detection on a new user.

On the basis of the above discussion, we conclude that *in general* the "tempting" procedure of enumerating all possible approaches is infeasible. However, there is one important exception. If the event space X can be reduced in size sufficiently, then the approach is implementable. This appears to be the case in the Wisdom and Sense algorithm. Because W&S relies on higher level auditing information (essentially information at the network level about logins) the size of the space X is severely restricted. Even in this restricted space, processing is a challenging task, and the W&S developers deserve praise for the clever algorithms that they use to process and encode the data. We decided not to pursue this method of restricting the scope of data to process statistically with NIDES for three reasons: (1) we felt that the expert system could do a fairly good job of detecting many types of unusual login activity, (2) we wanted NIDES to be capable of processing more detailed information about user activity, and (3) NIDES was designed to be extensible and allow the inclusion of modules such as W&S.

Our approach to the problems raised by the size of X was to assess the unusualness in each measure individually. (After we had assessed the unusualness of each measure individually, we summed these assessments to obtain an overall assessment of unusualness.) This approach dramatically lowers storage and the sample size requirements. The corresponding loss is that we cannot distinguish changes in the combinations of values of measures. For example, if a user only executes command "a" with file "b" and command "c" with file "d", then an intruder onto this user's account who executes command "a" with file "d" and command "c" with file "b" will not be flagged as anomalous. (This limitation is not inherent in the NIDES algorithm. Although

be times when x denotes activity over a very short period of time (i.e., seconds or minutes) and times when x denotes activity over much longer periods of time (i.e., hours of near inactivity).

This discussion demonstrates the following:

- The dimensionality of x is potentially very large. At any particular time there may be many different processes executing (or temporarily in abeyance). Each individual process can have many different files open. Often, each process can have executed numerous commands. Whenever audit records address only very limited single actions (such as is the case with Sun-Unix audit records), we would want x to span a reasonable period of time. There is considerable uncertainty concerning whether to measure time by the clock or number of audit records, and either way can lead to undesirable results.

- The space X is at least as complicated as any outcome space considered in the statistical literature. The vast majority of statistical literature deals with very simple outcome spaces (e.g., independent or serially correlated outcomes, each of a fixed constant dimensionality) that cannot be applied to X without considerable modification.

In defining the NIDES statistical component we found it useful to let X be the space of all prior behavior (extending backwards in time to the first actions of the user). However, we weighted the audit records within x so that the more recent audit records were much more influential in our decision making. Furthermore, we found that by defining a set of statistics (i.e., long-term profiles) that were easily updatable exponentially weighted sums, we were able to avoid much of the complexity in the space X. For example, we could update our statistics only by looking at the most recent audit record, without having to consider the relationships of the current audit record to previous audit records.

4.3 Q: How Has the Size of the Space X Affected the NIDES Statistical Algorithm?

One tempting approach to anomaly detection is to enumerate all of the possible outcomes, observe the empirical probability for each of those outcomes for the normal user, and then declare a recent outcome (i.e., an audit record or set of audit records) to be anomalous if it is one of the outcomes that historically has rarely been seen. Unfortunately, the size of the space X precludes this approach.

The size of the space X is huge. Consider only two measures — command name used and file name used. A typical user might use 100 different commands and have a file system with 1000 different file names (excluding temporary files that are generated

subject. We want to decide whether x is representative of the normal behavior of the user or whether x is anomalous (and therefore indicative of misuse). For example, Helman et al. (1992) define two stationary stochastic processes on the outcome space — $N(x)$, which denotes normal or legitimate transactions, and M(x), which denotes misuse transactions — that map values of x into probabilities or probability densities, and then later show the optimal test statistic for determining whether the particular value of x that was seen came from $N()$ or $M()$.

The ease in theoretically specifying that there is a space X belies the practical difficulties in actually defining X. (This difficulty directly affects the feasibility of implementing the Helman approach and has also influenced the NIDES statistical approach.) The following problems arise:

- Individual audit records typically do not contain complete values x within X. We observe a sequence of audit records, each of which contains limited information about user activity. Let us denote these audit records as r_t where t is the unique time stamp on the audit record. For example the value for r_t might denote that at the time that the last audit record was generated (i.e., time t) a particular command was being executed, a particular file was being accessed, the user was logged on from a particular terminal, and so forth. One method for specifying a value x is to let x be the complete past history of audit records or, perhaps more reasonably, all audit records generated over the last 10 minutes. That is, $x = \{r_t, r_{t-1}, r_{t-2}, r_{t-3}, r_{t-4}\}$, etc.

- A substantial number of the values of r_t are missing. For example, a particular audit record might not contain information about files being used at that moment, or about I/O, memory, or CPU usage. In addition, because a user can (intentionally or unknowingly) spawn a number of processes — all of which can be simultaneously using computer resources (even on a single machine) — and each audit record is tied to a specific process, each audit record gives only partial information about the totality of activity that is going on in a single user's account at the time of audit record generation.

- Because audit records from separate processes can be intermingled (as different processes swap in and out of memory or are actually simultaneously executed on different processors), user behavior as recorded on audit records can appear to be quite erratic. This type of behavior is not well explored in the literature (which usually assumes rather well-behaved time series or stochastic processes).

- Because there may be long periods without any audit activity and periods of time with very intense activity, it is not immediately apparent how many audit records should be included in x. For example, if x is defined in terms of clock time (say 5 minutes), there will be periods in which x is composed of thousands of audit records, and many other times when x is empty. On the other hand, if x is defined in terms of audit records (say a few hundred audit records), there will

- There were insufficient data on intrusion attempts to train a statistical approach.
- The typical (normal) usage of other users was not believed to be a reasonable surrogate for intrusion attempts to train a statistical approach.
- Future intrusions need not follow historical methods.

This decision was probably the single most important and influential decision in shaping the statistical approach. If we had been willing to model intrusive behavior, then we could have applied the Neyman-Pearson Lemma (and the decades of statistical research that followed from this lemma) to develop an optimal statistical approach. The Neyman-Pearson Lemma (proven in the 1930s by two of the founders of modern statistical theory) states that if there are two known probability distributions, then the optimal statistical test for distinguishing between these two distributions is based on the ratio of their values. For example, let x denote an outcome (derived from audit records), let $n(x)$ denote the probability of that outcome for the host user, and let $m(x)$ denote the probability of that outcome for the intruder. Suppose that we want to determine the set of outcomes X such that X occurs very rarely for the host user (say no more than 0.1% of the time) but much more frequently for the intruder. The optimal selection for X is the set of values of x with the highest values for the ratio $m(x)/n(x)$. That is, we order all values for x according to the ratio of $m(x)/n(x)$ and add x values into X until $n(X)$ equals 0.1%. A restatement of this lemma in a Bayesian framework can be found in Helman et al. (1992).

If we had chosen to model intrusive behavior, then by access to the Neyman-Pearson Lemma we could have developed an optimal approach for detecting intrusive behavior. This would have allowed use of many standard statistical approaches (such as discriminant analysis, Hotelling's T^2 tests) that stemmed (either directly or indirectly) from this lemma. Instead, we were faced with the task of defining a region X without reference to assumed intruder behavior, but rather based on the fact that such behavior was unusual for the user (whether or not it would be unusual for the intruder).

4.2 Q: What Does the Space X of Outcomes Look Like? How Has the Unusual Aspects of this Space Influenced the NIDES Statistical Component?

Statistical intrusion approaches accept audit record inputs that define the outcomes of subject behavior. These outcomes define a space, which we have denoted as X. A member x belonging to X is the particular behavior that has been exhibited by a

Chapter 4

Answers to Specific Questions about Features of the NIDES Statistical Algorithm

The NIDES statistical approach has evolved over many years. This evolution has been motivated by a mixture of theoretical considerations, practical considerations (dealing principally with issues of computing speed), and practical experience in exercising the algorithm. Here, we identify some of the major decisions that shaped the statistical approach and describe some of the specific features of the algorithm. We have adopted a question-and-answer format, where the questions deal with different features of the NIDES statistical approach, and the answers offer insight into why those features were incorporated. Occasionally, we group a set of questions that have closely related answers. We have also formulated a number of the questions (particularly the questions that appear early in this section) so that they illuminate the relationship between NIDES and the approach of Helman et al.[9]

4.1 Q: Is the NIDES Statistical Approach Based on an Assumed Intruder Behavior?

A decision made early in the NIDES development process is that we should not assume any knowledge of how intruders would compromise system security (or how a user would alter his or her behavior to compromise system security). All knowledge of this nature was to be incorporated into the expert system. We made this decision for four principal reasons:

- The expert system is typically more accurate in detecting known intrusion approaches.

3.12 Criterion 12. Does the Method Provide the Security Officer with Adequate Information to Conduct an Inquiry?

By their very nature, statistical approaches to anomaly detection will raise false positive alerts. That is, users do change their behavior over time — and sometimes abruptly so. Statistical algorithms will detect these changes and issue alerts even though no misuse activity is actually transpiring. Because the vast majority (i.e., 99% or more) of activity on a typical computer system will be legitimate, the number of false positive alerts will tend to outnumber the true positive alerts. A good deal of the system security officer's time will therefore be devoted to investigating false positive alerts. The statistical procedure must be capable of providing the security officer with information that can be used to identify the reason why the alert was issued. This will tend to greatly reduce the time involved in the security officer's determination of whether misuse has actually occurred.

The requirement for "understandability" may restrict the type of statistical procedures that can be applied. For example, factor analytic procedures tend to transform the space of measures in complicated ways, and it can become quite difficult to unscramble the statistical results in such a way that they can be explained to the security officer.

Because of the need for understandability, the NIDES statistical component uses the concept of long- and short-term profiles, which are probability distributions for categorical measures. Although the presentation format for these probability distributions still requires improvement, the concept of frequency distributions is easily grasped. For example, the security officer can see how frequently different commands have been issued in the past and which commands have been more recently issued. It is relatively easy then to identify those commands that are now being issued, but have not been issued in the past (or were issued much less frequently in the past) as well as those commands that were frequently issued in the past but are now conspicuously absent. The use of relatively simple individual measures (e.g., commands used, file names, hosts used, number of access denials) makes the results more comprehensible.

as non-A, that unauthorized activity must much more closely resemble the typical activity of other users than the activity of user A. In the event that the intrusive activity does not closely match either user A's activity or the activity of other users, statistical approaches such as discriminant analysis might fail to classify the activity as not belonging to A.

The NIDES methodology continuously compares a user's recent activity with his or her long-term profile. It is in this sense self-referential and independent of the activity of other users on the system. As such, it makes no assumption as to the nature of the difference between legitimate and unauthorized activity.

3.11 Criterion 11. Does the Method Require Assumptions about the Distribution of Intrusive Behavior?

Because data on actual and simulated intrusions are scarce, and a number of statistical procedures have been proven to have optimality properties when comparing two known distributions, it is tempting to make assumptions concerning the distribution of intrusive behavior. These assumptions are typically of one of two types. The first is an assumption that intrusive behavior always tends towards larger (or sometimes smaller) values of a particular measure of resource consumption (e.g., CPU time, I/O, file accesses, "finger" commands). The second is an assumption that intrusive behavior is uniformly distributed across all possible behaviors.

Although these assumptions allow "optimal" tests to be derived, unless the correctness of these assumptions is verified, the optimality of these tests must be severely discounted. In fact, most "reasonable" statistical tests can be shown to be optimal or near optimal versus some assumed distribution of intrusive behavior. This can be done by finding the set of behaviors that would result in the declaration of an anomaly and then assuming that intruder behavior is concentrated on those outcomes. (Unfortunately, the math involved in making such assessments is quite complicated.)

The NIDES statistical algorithms were not postulated with a specific distribution of intrusive behavior in mind, nor have we reverse-engineered the set of intrusive behaviors for which our statistic is optimal. However, the form of our test statistic (e.g., similar to a chi-square statistic) suggests that we spread our detection power over many possible intrusive behaviors and so would do particularly well versus a flat or uniform distribution for intrusive behavior.

3.9 Criterion 9. Does the Method Require the Existence of Simulated or Actual Intrusions?

Many statistical classification methods such as discriminant analysis or some neural networks require that training sets exist and the audit data in the training set be preclassified. For example, one portion of the training set might consist of audit data from users who are known not to be misusing the computer system and who own accounts that have not been subjected to intrusions. The other portion of the training set might consist of audit data gathered during simulated or actual misuse or intrusions. By comparing the "normal" and "intrusive" audit records in the training set, the statistical method can determine what features in the audit data best classify the data as normal or intrusive.

Unfortunately, data sets of simulated or actual intrusions are not widely available. (If such data sets had been available, we would probably have included a statistical classification method in NIDES.) A researcher who proposes to use a classification approach that requires the existence of simulated or actual intrusions should have a good idea where such data would come from or how to generate it. In addition, the ability of the statistical method to detect intrusions will generally be limited by the type of intrusions that it is trained on. That is, it will be able to detect new types of intrusions only to the extent to which they share features with the types of intrusions in the training set. Therefore, the generality and completeness of the simulated or actual intrusions in the training set should be assessed.

3.10 Criterion 10. Does the Method Develop Its Assessment Based on Differences between Users?

Because training sets of simulated or actual intrusions are not typically available, the researcher might propose to apply the statistical classification method to a training set consisting only of normal users. For example, to develop an algorithm for identifying abnormalities in User A, one portion of the training set might consist of user A's audit data and the other portion consist of a mixture of audit data from other users (which might be identified either as "non-A" or by their individual user IDs). The statistical method would then be trained to classify a new audit record either as A or as non-A.

Statistical approaches that distinguish normal users from one another might also be capable of identifying misuse or intrusive activity in an audit stream. We believe that such methods are worth investigating. However, it is not clear that such methods would actually be able to detect unauthorized activity. For example, for there to be a high probability that unauthorized activity in User A's account would be identified

process unacceptably. The NIDES statistical component does not require human intervention to update profiles.

3.7 Criterion 7. Does the Method Allow for Multivariate Statistical Inference?

It is our belief that much of the ability to detect interactions rests in the interaction of different aspects of behavior. For example, a set of particular commands may be associated with a particular host computer or with action on a particular set of files. The statistical approach should therefore be able to extract additional information about the normality of different actions taking place essentially simultaneously (e.g., invoking commands and accessing files).

The use of highly dimensional multivariate data involves many difficulties that affect the ability of the statistical approach to satisfy other criteria. For example, the multivariate cross product of only three measures, with only 100 categories each, results in a matrix of 1,000,000 possibilities. Even though the matrix would probably be sparsely populated, storing and processing such a matrix would tend to make the statistical algorithm too slow and the storage requirements too large. In addition, development of stable probabilities for the matrix cells would require long training periods — perhaps sufficiently long that behavior would change before training was completed.

The NIDES approach makes limited use of multivariate data, and we consider this to be one of the statistical component's primary limitations. However, we have developed the theoretical approach necessary to incorporate multivariate data (at least to some extent), and could incorporate this approach in future versions of NIDES.

3.8 Criterion 8. Does the Method Require the Existence of Defined Jobs or Sessions?

Computing used to be defined in terms of batch jobs. Now most processing takes place in continuous sessions, which may last for many hours. (In the extreme case, the user may never log off.) The statistical approach should not require that there be "jobs" or "sessions" or their equivalents, but rather should make assessments about data from a continuous stream of audit records. The lack of a natural time unit of analysis (that is, job or session) is directly responsible for the use in NIDES of exponentially weighted statistics.

user have his or her own profile. This requirement affects both the time necessary to update profiles (since there are many profiles to update) and memory requirements. Profiles must be capable of being stored in a reasonable amount of space, or an excessive amount of time will be spent swapping profiles in and out of memory. Currently, NIDES requires about 100kb per profile in the Sun-Unix environment, which has a rich set of measures available. We suspect that the NIDES profiles could be stored in less space when using audit record data from other computer systems.

In some environments job classifications may be tightly prescribed, and individuals in the same job classification may use the computer in essentially the same way. In this case it may be possible to develop useful group profiles, and individual profiles may not be necessary. A statistical approach used in such an environment would not need to allow for individual user profiles, because group profiles would be sufficient. However, because NIDES was developed as a general-purpose intrusion-detection system, we developed a statistical component that could efficiently process individual profiles.

3.6 Criterion 6. Does the Method Allow for Profiles to Periodically Update without Human Intervention?

User profiles are constantly changing as the legitimate work of the user changes. Therefore, it is necessary that the statistical procedure be capable of updating user profiles at least once daily. Because there may be thousands of user profiles, efficiency in updating is important.

If we assume that updating should occur once per day, and that during the 10 business hours of the day the statistical system will be too busy processing audit records to update profiles, then the profiles will need to be updated during the remaining 14 hours (i.e., 50,000 seconds) of the day. If there are 2000 user profiles (i.e., a decently large system), each profile will need to be updated within 25 seconds. (With fewer users the profile updating time can take longer, and with more users the profile updating time will need to be less.) If we put in a safety margin of 50%, then profiles for a 2000 user system should each update within 12.5 seconds. While most statistical algorithms are capable of updating in this time frame, some may require more than this amount of time. The NIDES statistical component can update all profiles for an individual user in a second or so, and consequently if we initiate updating at midnight, we are typically finished within tens of minutes later.

It is also important that the updating procedure not require the guidance of a trained statistician, but rather be completely automated. Some statistical procedures, such as cluster analysis, typically require the intervention of a statistician to obtain decent results. A statistician will not typically be available when updating is required, and even if he or she were available, human intervention would slow the updating