A Fast Computer Intrusion Detection Algorithm Based on Hypothesis Testing of Command Transition Probabilities

William DuMouchel
AT&T Labs-Research
180 Park Avenue
Florham Park, NJ 07932
dumouchel@research.att.com

Matthias Schonlau
AT&T Labs-Research and
National Institute of Statistical Sciences
PO Box 14006
Research Triangle Park, NC 27709-4006
schonlau@research.att.com

Abstract

This statistical method compares in real time the sequence of commands given by each user to a profile of that user's past behavior. We use the Fisher score statistic to test the null hypothesis that the observed command transition probabilities come from a profiled transition matrix. The alternative hypothesis is formed from a principal components analysis of historical differences between the transition probabilities of all other users and those of the user being tested. The calculations can be structured so that only a few dozen arithmetic operations are needed to update an online test statistic after each submitted command. The theoretical statistical properties of the test, such as false positive and false negative rates, are computable under the assumptions of the markov process model. Based on a population of 45 research users on a single computer, test data from each user are used to challenge the profile of every user. The test had sufficient statistical power to successfully discriminate between almost every pair of users based on a sample size equivalent to a single day's usage of an average user.

Description of Statistical Methodology

Introduction. In computer intrusion detection one attempts to identify unauthorized accesses to computer accounts. There are two main approaches to intrusion detection: pattern recognition and anomaly detection. Pattern recognition is the attempt to recognize general "attach signatures" that stem from known attacks such as exploiting a software bug. The approach has the disadvantage that it cannot defend against previously unknown software bugs, or any unauthorized user with the knowledge of the account password. Anomaly detection, on the other hand, attempts to identify an unauthorized user by identifying unusual usage of the computer. Usually, for each user a historical profile is built and large deviations from the profile indicate a possible intruder. Therefore it is also referred to as the profile based approach. Intrusion detection systems like IDES (Lunt et al. 1992), NIDES and Emerald (Porras and Neumann 1997) use both approaches, presumably because neither one is uniformly superior to the other. In this paper we only consider the anomaly detection approach. This approach lends itself to a statistical treatment. Ryan et. al. (1998) suggested that each user on a computer system leaves a "print" that could be captured by training a neural network with historical data. When for new data from any user the neural network predicts that the data is more likely to stem from another user in the historical data, then an alarm for a possible intrusion is raised. Forrest et al. (1996) consider anomalies for unix processes (such as ftp, or root) rather than for users. In this paper we propose a test for anomaly detection based on hypothesis testing. Since users (including the root user) and system processes (such as ftp) both generate command - level data, we are able to test anomalies for unix processes and users simultaneously.

Command Transition Probabilities. Our method is based on comparing the sequence of each user's commands to a stored profile describing the probability distribution of that user's command sequences. Each command is one of a fixed number K of possible commands or command groupings. We represent each user's historical data in terms of a transition matrix of command probabilities:

$$p_{jku} = P(\text{Next Com.} = k | \text{Prev. Com.} = j, \text{User} = u) \tag{1}$$

The commands are arbitrarily numbered as $j, k = 1, ..., K$ and we assume that historical data are available for $U+1$ users, arbitrarily numbered $u = 0, 1, ..., U$. We focus on User $u = 0$ as the user whose commands are being monitored and tested for unusual behavior at any given time, but the historical data from the other U users play a role in the test procedure for user 0.

Smoothing Historical Transition Probabilities

The probabilities (1) must be reliably estimated, with all $p_{jku} > 0$. Thus we smooth the observed proportions from the training period. Denote by N_{jku} the raw count of transitions from command j to command k for user u during the training period. This array will have many elements equal to 0 and many of its nonzero elements may be so small that they are statistically unreliable.

For some small $\epsilon > 0$ (we use $\epsilon = 0.001$), let

$$N_{.ku} = \sum_j N_{jku}$$

$$N_{..u} = \sum_k N_{.ku}$$

$$q_{ku} = (N_{.ku} + N_{..u}\epsilon)/N_{..u}(1 + K\epsilon)$$

The q_{ku} are the marginal probabilities of command k for user u, modified using ϵ so that they are all positive. The transition probabilities p_{jku} are weighted averages of q_{ku} and the raw proportions $N_{jku}/N_{j.u}$, namely

$$p_{jku} = [(N_{jku}+M_{ju}q_{ku})/(N_{j.u}+M_{ju})+\epsilon]/(1+K\epsilon) \quad (2)$$

where $N_{j.u} = \sum_k N_{jku}$ and M_{ju} are weights chosen to be large if the raw transition frequencies are not significantly different from the marginal frequencies, but near 0 if the raw transition frequencies are very reliably different than the marginal ones. Specifically, let

$$M_{ju} = 1/\max\{.0001, [\sum_k (N_{jku} - N_{j.u}q_{ku})^2/ N_{j.u}q_{ku} - K + 1]/N_{j.u}(K-1)\} \quad (3)$$

The use of equations (2) and (3) is inspired by an empirical Bayes model in which it is assumed that each vector $(p_{j1u}, ..., p_{jKu})$ was generated from a Dirichlet prior distribution (O'Hagan 1994, Ch. 10) with mean vector $(q_{1u}, ..., q_{Ku})$ and probability density proportional to $\prod_k {p_{jku}}^{M_{ju}q_{ku}-1}$.

Hypothesis Testing Framework

Suppose that User 0 is logged on and has generated a sequence of $T+1$ commands $C_0, C_1, ..., C_T$. There is the possibility that these commands are being generated by someone other than User 0, and let the unknown true transition probabilities of this sequence be

$$\pi_{jk} = P(C_t = k|C_{t-1} = j) \qquad t = 1, 2, ..., T \quad (4)$$

The corresponding transition counts for this user's test data are n_{jk}, where $\sum_{j,k} n_{jk} = T$. Let $n_{j.} = \sum_k n_{jk}$. By definition, $E[n_{jk}|n_{j.}] = n_{j.}\pi_{jk}$. We want to test the null hypothesis

$$H_0: \quad \pi_{jk} = p_{jk0} \qquad j = 1, ..., K; \; k = 1, ..., K \quad (5)$$

Statistical hypothesis testing is a procedure in which a decision maker prespecifies a computable test statistic whose distribution is supposed to be completely known assuming that a null hypothesis H_0 is true. This allows the false alarm probability to be computable in advance for a decision rule that rejects H_0 whenever the test statistic exceeds a fixed value.

The standard test statistic for this situation is the log likelihood ratio statistic, namely

$$LR = 2\sum_j \sum_k n_{jk}\log(n_{jk}/n_{j.}p_{jk0}) \quad (6)$$

The range of j in the above summation is over the J values of j for which $n_{j.} > 0$. If T is very large and the assumptions (4 - 5) are true, then LR will have an approximate chi-squared distribution with $J(K-1)$ degrees of freedom, so that LR could be compared to the percentiles of this distribution to assess H_0. Unfortunately, T will rarely be large enough for this distributional assumption to be even approximately true. The usual rule of thumb is that every value of $n_{j.}p_{jk0} > 1$, and that most $n_{j.}p_{jk0} > 5$. This would imply that every $n_{j.} > 3K$, yet many rarely occurring commands would have many fewer occurrences than that in the test data. Another problem with using the test statistic (6) is that this is an omnibus test with power against all possible alternatives to (5), which is likely to waste statistical power testing against unlikely or nonsensical alternative sets of transition probabilities. Therefore we will construct an alternative hypothesis with fewer degrees of freedom, using the historical data from all users as a guide to which alternative transition probabilities are plausible.

Alternative Hypothesis. Suppose that the test data are being generated by the transition probabilities

$$H_1: \pi_{jk} = \pi_{jk}(\beta) = p_{jk0} + \sum_{u=1,U} (p_{jku} - p_{jk0})\beta_{ju} \quad (7)$$

where $\beta = (\beta_{11}, ..., \beta_{KU})$. Instead of considering all alternatives to H_0, (7) focuses on directions in the space of transition probabilities that the historical data confirm are occupied by one or more other users. The expression (7) becomes identical to H_0 if all KU elements of $\beta = 0$, so H_1 might also be stated as $\beta_{ku} \neq 0$ for some (k, u). However, (7) still has the disadvantage of too many degrees of freedom because many of the users may have similar historical probabilities so that the vectors $(p_{j1u} - p_{j10}, ..., p_{jKu} - p_{jK0})$ may be highly collinear for many values of u.

Principal Components Regression. The dimensionality of the alternative hypothesis is reduced by choosing linear combinations of user deviations from p_{jk0} that have maximum variance and are uncorrelated. See DuMouchel and Schonlau (1998) for more details of the statistical theory behind this approach. First define the matrices X_j:

$$X_{jku} = (p_{jku} - p_{jk0})/\sqrt{p_{jk0}}$$

Then let Z_j be a $K \times U^*$ matrix consisting of the first $U^* < \min(U, K)$ principal components of X_j. Take uncentered principal components, so that $tr(Z_j'Z_j) \approx tr(X_j'X_j)$.

Test Procedure

After reducing the dimensionality in this way, the corresponding Fisher score statistics (Stuart and Ord 1991, Ch. 25) are defined as follows, where $v = 1, ..., U^*$:

$$Y_{vjk} = Z_{jkv}/\sqrt{p_{jk0}} \quad (8)$$

$$S_{jv} = \sum_k n_{jk}Y_{vjk}$$

$$V_{jv} = \sum_k p_{jk0} {Y_{vjk}}^2$$

The principal component scores S_{jv} have moments

$$E[S_{jv}] = 0 \tag{9}$$

$$Var(S_{jv}) = n_{j.} V_{jv} \tag{10}$$

$\mathrm{Cov}(S_{jv}, S_{j'v'}) = 0$ for all combinations $j \neq j'$ or $v \neq v'$. Note that if the $U^* K^2$ subscores (8) are stored, then only U^* additions are required to update the entire matrix S_{jv} after each observed transition $(C_{t-1} = j, C_t = k)$.

Principal components chi-squared. The test statistic

$$S^2 = \sum_{j,v} ({S_{jv}}^2 / n_{j.} V_{jv}) \tag{11}$$

has an approximate χ^2 distribution with $df = JU^*$ under H_0, or, using the Wilson-Hilferty (1931) transformation,

$$S^* = [(S^2/df)^{1/3} - 1 + 2/9df](9df/2)^{1/2}$$

In our example, $U^* = 5$ principal components are used for each previous command j.

Data and Results

To evaluate the method presented in the previous section, we compare test data to training data (profiles) for pairs of users. Ideally, an alarm should always be raised except when a user is tested against his or her own profile. To establish user profiles, we use historical data from usage on our local unix machine. The data (user names and commands) are extracted from output of the unix `acct` auditing mechanism and consist of user names and commands only (without their arguments). Some commands recorded by the system are implicitly generated and not explicitly typed by the user. For example, each execution of the .profile file or a make file generates commands contained in these files that are also recorded in the data stream. We use test data separated in time from training data according to the following cross-validation scheme involving four separate time periods.

Experimental Design. Data were collected from our local population during four separate time periods approximately a month apart. During each time period, the first 1000 command transitions by each user are included in the study. There were 45 users with that amount of data available from all four time periods. We form four separate replications of our study by considering in turn each of the four time periods as the test period and the other three time periods as historical training periods for collecting user profiles. Within each replication, we test each user's test data against each of the profiles in the historical data and record the standardized test statistic S^*. Within each set of 1000 test commands, we compute the test statistic for each of 10 sets of 100 commands, so the basic unit of study is observation of 100 commands from the user being validated. When the standardized test statistic S^* exceeds a threshold value, an alarm is raised. Depending on the chosen value of the threshold, different rates for false positives (false alarms) and false negatives (missing alarms) can be obtained. In the following we investigate the tradeoff between false negatives and false positives by choosing different thresholds.

Results. Figure 1 focuses just on the false alarm problem (comparing each test user to the same user's profile) and looks at the test statistic as a function of the number of commands N. Each of the 45 curves in Figure 1 corresponds to one test data stream. The two horizontal lines in Figure 1 correspond to thresholds of $S^* = 10$ and of $S^* = 100$ respectively. Out of 45 users, only 2 ever exceed the threshold of 100. The vast majority of users maintain $S^* < 10$. Under the

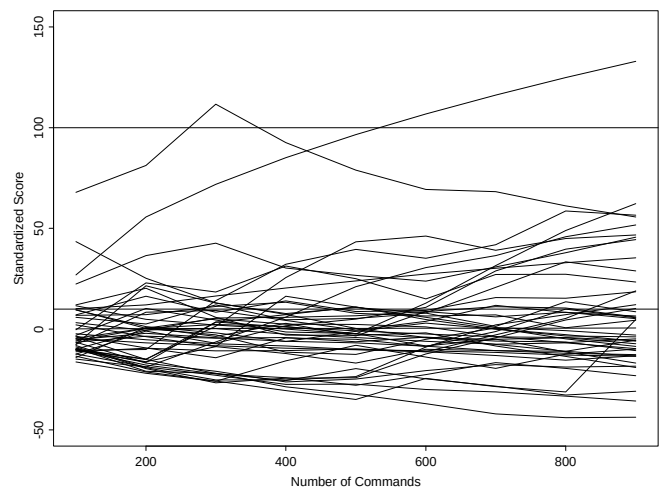

Figure 1: Time plot of standardized score comparing each user's test data with their own training data profile. Training Periods = 1−3. Test Period = 4. Number of Principal Components $U^* = 5$.

Null hypothesis, S^* should have a standard normal distribution which would suggest a small threshold (e.g. 3). Our data show higher false alarm rates than this theory predicts, which we attribute to the fact that our model does not accommodate users' behavior changing over time, and some of our users changed significantly.

Figure 2 shows, for each of the 45×45 test-data versus a profile pairs, the median value of S^* for blocks of $N = 100$ commands within that pair. (Note that actually $20 + S^*$ is plotted to allow for the logarithmic scale on the vertical axis. A vertical line is drawn at $S^* = 0$). Ideally, the median test score against a user's own historical profile (denoted by "+" in Figure 2) should be lower than all of the test scores run against other people's profiles (denoted by "."). Another way of saying this is that a user should only be able to break into his/her own account without causing an alarm to be raised. As we can see in Figure 2, the block medians

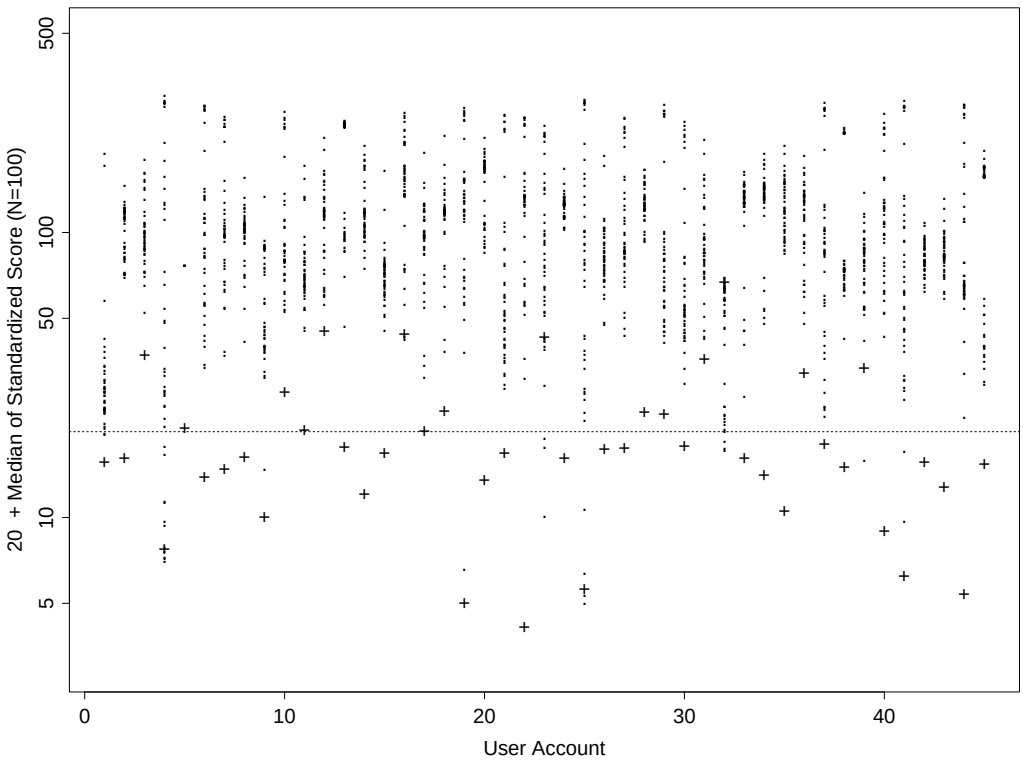

Figure 2: Medians of standardized scores for blocks of 100 test data commands, compared to each training data. "+" denotes comparison to own training data, "." denotes comparison to other users. Training Periods = $1-3$. Test Period = 4. Number of Principal Components $U^* = 5$. A horizontal line is drawn at $S^* = 0$.

discriminate almost perfectly between a user's own and other profiles.

As the threshold where an alarm is raised changes, different rates of false alarms (false positives) and missing alarms (false negatives) can be obtained. Figure 3 and Figure 4 show the tradeoff between these false positive and false negative rates for all four time periods based on blocks of 100 and 900 commands, respectively. Each time period is used as testing data with the remaining three time periods serving as training data. The lower left corner represents the ideal scenario: no false alarms and no missing alarms. Note that both axes are presented on a logarithmic scale. From Figure 3 we can tell, for example, based on 100 transitions, at a false positive rate of 5% we get a corresponding false negative rates between about 15% - 60%. Based on 900 transitions (Figure 4), the false negative rates drop to $5\% - 31\%$. These numbers are quite high and also indicate that there is a considerable variability from time period to time period. The variablility in Figure 4 is in part due to the relatively small sample size: an alarm for 2 out of 45 blocks of 900 commands amount to about 5% false alarms.

Discussion

Ryan et. al. (1998) use a neural network approach and test classification errors based on 10 users. They have 11 successive days of data, 8 of which are used for training. One of the users only had little data. They report a false alarm rate of 7% and 4% missing alarms. Our test is more challenging in that we test with more users, and because there is a gap in time between historical and test data. Also, their decision criterion seems to assume that the intruder would be one of the other users in their training data. On the other hand, unlike them, we excluded users with very low account usage.

In order for an intrusion detection tool to be useful, the false alarm rate needs to be low – otherwise alarms tend to be ignored. To that extent a false alarm rate of 5% still seems high. Perhaps extending the length of the training period will make the profiles more robust to changes in user behavior. One possible way to improve the markov model is by consolidating series of cascading commands (generated, for example, by a makefile) into single (meta) commands, or by considering the next command conditional on more than one previous command. Note that our theoretical false alarm probabilities ignore the problem of multiple test-

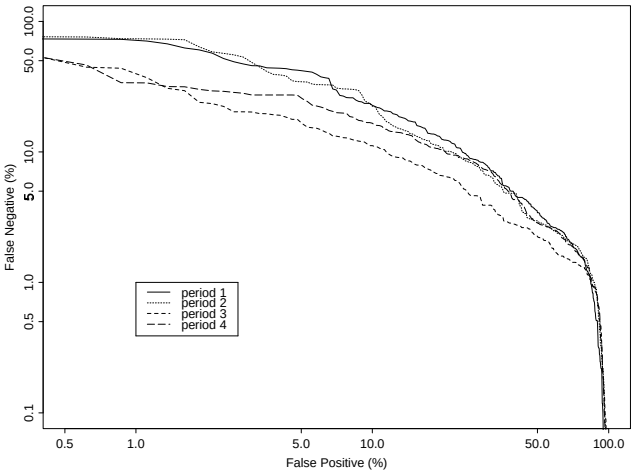

Figure 3: Tradeoff between false positive and false negative rates for statistic (11), treating every block of 100 commands as a separate experiment. Each of four periods is used as test data with the respective other three periods being used as training data. Number of Principal Components $U^* = 5$.

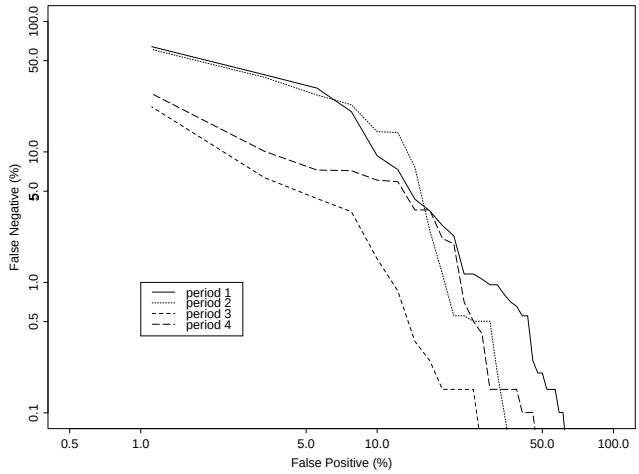

Figure 4: Tradeoff between false positive and false negative rates for statistic (11) based on 900 commands for each user. Each of four periods is used as test data with the respective other three periods being used as training data. Number of Principal Components $U^* = 5$.

ing, in which repeated testing of the same null hypothesis as time goes on increases the chance of a false alarm rate. This problem, common to most control-chart like procedures, seems to be a less important cause of excessive false alarms than our failure to model how users tend to change their profiles over time.

A major strength of the approach presented is its speed. Only a few dozen operations are needed for updating the test statistic, and preliminary calculations indicate that it will be easily possible to implement this procedure in real time. The amount of storage required for the procedure is relatively large. Based on 5 principal components and 100 command categories, 50500 single precision numbers need to be stored for each user.

Because of space constraints, we report here only on the behavior of the S^* statistic and defer comparisons for varying numbers of principal components and different test statistics. See DuMouchel and Schonlau (1998) for description of a comparative study of different test statistics and numbers of principal components to use with this method.

We need to perform further investigation of the optimal number of principal components to take. We will also investigate the effect of adding known intrusion signatures as profiles in the training data to make the principal components methodology more sensitive to such attacks. We also expect to be able to use this procedure to increase our understanding of local system usage.

References

DuMouchel, W. and Schonlau, M. (1998). A Comparison of Test Statistics for Computer Intrusion Detection Based on Principal Components Regression of Transition Probabilities. In *Proceedings of the 30th Symposium on the Interface: Computing Science and Statistics*, (to appear).

Forrest, S., Hofmeyr, S., Somayaji, A., Longstaff, T. (1996). In *Proceedings of the 1996 IEEE Symposium on Security and Privacy.* IEEE Computer Society Press, Los Alamitos, CA, pp. 120-128.

Lunt, T. Tamaru, A., Gilham, F., Jagannathan, R., Neumann , P., Javitz, H., Valdes, A., Garvey, T. (1992). A Real-Time Intrusion Detection Expert System (IDES) - final technical report. Computer Science Library, SRI International, Menlo Park, California.

O'Hagan, A. (1994). Kendall's Advanced Theory of Statistics, Vol. 2B: Bayesian Inference, New York: John Wiley.

Porras, P., and Neumann P. (1997). EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disurbances. In *Proceedings of the National Information Systems Security Conference.* (to appear).

Ryan, J. , Lin,M., and Miikkulainen, R. (1998). Intrusion Detection with Neural Networks . In Jordan, M. I., Kearns, M. J., and Solla, S. A. (editors), Advances in Neural Information Processing Systems 10 (NIPS'97, Denver, CO). Cambridge, MA: MIT Press.

Stuart, A. and Ord, J.K. (1991). Kendall's Advanced Theory of Stististics, Vol. 2: Classical Inference and Relationship, New York, John Wiley.

Wilson EB, Hilferty MM (1931). The distribution of chi-square, In *Proc. Nat. Acad. Sci.*, Wash. DC, 17, 684-688.