Intrusion Detection Based on Structural Zeroes

Martin Theus
AT&T Labs-Research
180 Park Avenue
Florham Park, NJ 07932

Matthias Schonlau
AT&T Labs-Research and
National Institute of Statistical Sciences
PO Box 14006
Research Triangle Park, NC 27709-4006

Abstract

A method for computer intrusion detection is proposed. It uses command-level data and is based on structural zeroes of user/command contingency tables. More specifically, it is based on commands that have non-zero entries for very few users. Hence it deals with a portion of the data that statistical methods usually ignore. The importance of structural zeroes in intrusion detection is demonstrated graphically. The proposed method successfully discriminates among 45 users based on their usage-data on our local research machine. Based on sets of 100 commands this method generates no false alarms at the cost of about 10% missing alarms.

1 Introduction

Since computer intrusion detection was first investigated (Anderson, 1980) much has happened. Several intrusion detection products are available commercially (e.g. Netranger, 1998, Emerald, 1998) and a number of academic institutions have research teams in this area (e.g. COAST project at Purdue University (1998), Intrusion Detection for large networks (U.C. Davis Research Group, 1998), Computer Immune Systems (University of New Mexico, 1998)).

There are two main approaches to intrusion detection: misuse detection and anomaly detection. Intruders often gain access to computer systems by exploiting software bugs. In misuse detection one tries to detect such attempts by recognizing attack patterns corresponding to known software bugs. The difficulty here is that not all software bugs are known, and that it is not useful once the intruder has already gained access to the system. The second approach, anomaly detection, attempts to detect deviations from past computer usage and flags them for investigation. Neither approach appears to be uniformly superior over the other. The approach presented here falls in this second category.

Our data consist of user names and commands (e.g. `ls, man, java`) from a UNIX operating system. Many cells of the user/command contingency table are empty. Since most users are only aware of some subset of the set of distinct commands or choose never to use commands they are aware of, most empty cells are structural zeroes (as opposed to sampling zeroes). Statistical techniques (e.g. principal components, regression analysis, etc.) generally assume that all combinations of variables/ observational units are feasible and can thus take non-zero values. While such techniques still function with large number of zero cells, they do not incorporate structural information.

Too many empty cells may also lead to estimation problems. For example, it is difficult to estimate transition probabilities from one command to a second command when the first command is never observed. Therefore it is tempting to exclude commands that are used by very few users from further consideration. DuMouchel and Schonlau (1998), for example, use only the 100 most frequently used commands (about 26% of all commands used by a group of 45 users) and combine all remaining commands into one category.

The outline of this article is as follows. In Section 2 we further motivate the importance of structural zeros. Section 3 introduces a heuristic test statistic based on this idea. Section 4 describes an experiment to test the statistic and gives results. Section 5 concludes with a discussion.

To facilitate explanations in later sections we discuss at this point our data and make some definitions.

Data

Command level data consisting of user names and commands (without arguments) were generated from output of the UNIX `acct` auditing mechanism. 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.

During two separate time periods (training and test data) the first 1000 commands for each of 45 users are recorded. For some of the analysis reported below, we split each set of 1000 commands into 10 replications of 100 commands.

Other sources of data (e.g. packet data) are conceivable. In this paper we are not concerned about what source of data would be most appropriate; rather we take the stance that we have to deal with the data available.

Definitions

A *unique command* is a command that is used by only one user in a training data set. A *rare command* is a command that is used by only a few users in a training data set. We further make the distinction between the *training data* which establishes, for example, how rare commands are and *testing data* which is used to test statistics. We denote the total number of users in any data set by U.

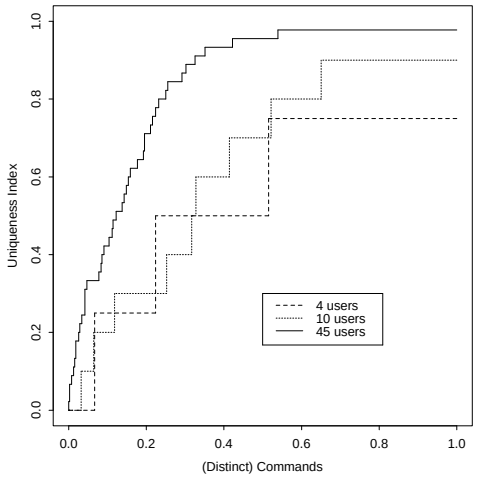

Figure 1: Stepplot of Uniqueness Index (1− Fraction of Number of Users) versus Distinct Commands for 4, 10 and 45 users.

2 Motivation

In this section we explore structural zeroes in our training data, that is, we look at how rare or unique commands are.

We first look at the question of what fraction of commands are used by only one user, used by exactly two users, etc. for a certain community of users. It is then interesting to plot the the fraction of commands used by exactly i users versus i. To overlay plots for a varying sizes of user communities, we standardize and use the fraction of users i/U instead. This plot is shown in Figure 1 for user communities of size 4, 10 and 45. Almost half of all commands are unique, in the sense that they are only used by one out of the 4, 10 or 45 users, respectively.

While Figure 1 shows that there are many unique and rare commands, it is unclear whether the unique/rare commands account for a large proportion of the total commands. Instead of the fraction of distinct commands, Figure 2 uses the fraction of the total frequencies on the horizontal axis. Two conclusions can be drawn from Figure 2: First, the percentage of the data consisting of commands used only by a certain percentage of users is approximately the same regardless of whether the user community consists of 4, 10 or 45 users. Second, since the lines are almost on the 45 degree line, the percentage of the data consisting of commands used only by a certain percentage x of users is approximately that percentage x. For example, about 20% of the command data contain only commands that are used by at most 20% of the users (those with a high uniqueness index in Figure 2).

The commands can be grouped such that each group contains all commands that are used by exactly i users ($1 \leq i \leq U$). We assign an ID to each command such that commands from groups with rare commands are assigned lower ID's than commands from groups with less rare commands. The order within a group is arbitrary. When plotting the command ID versus their order in the audit stream the usage pattern of unique/rare commands emerges. Such a plot is shown in Figure 3 for each of 10 users. Plots of different users are clearly distinguishable from each other. The fourth, seventh and the tenth plot are more regular than the other ones. They belong to UNIX processes (rather than to human users).

The same plot can be seen in Figure 4 for 45 users. (Figure 3 corresponds to the first 10 columns of Figure 4). While the time series aspect is now difficult to make out because each column is very narrow, the column patterns are clearly distinct. A classification of users based on which commands are used and how often they are used appears promising.

3 Intrusion Detection by Uniqueness

We are about to define a test statistic that depends on two quantities, W_{ij} and U_j. These quantities are esti-

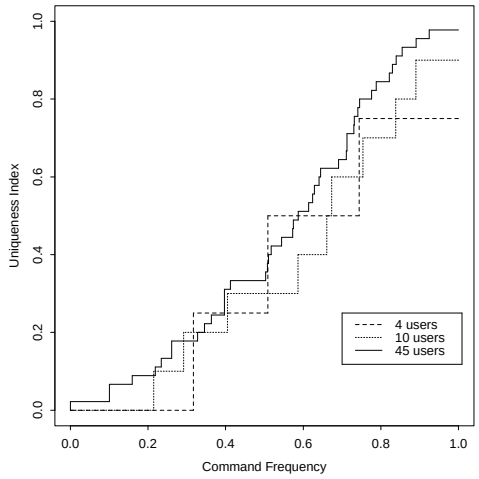

Figure 2: Stepplot of Uniqueness Index (1− Fraction of Number of Users) versus total Command Frequency for 4, 10, and 45 users.

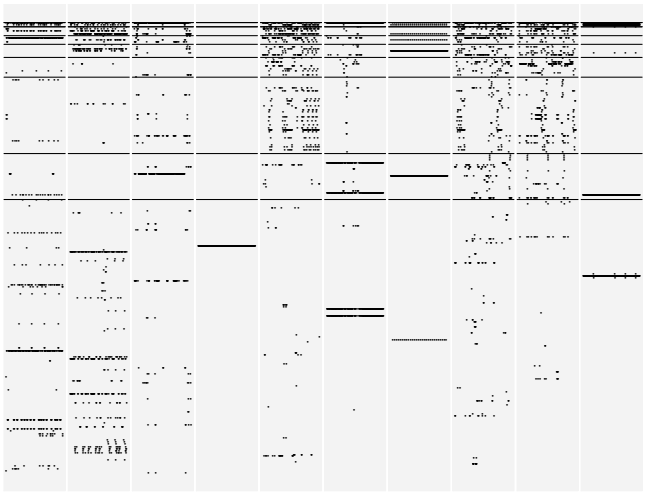

Figure 3: Plots of Command ID vs Command Order in the Audit Stream for Each User. Commands Are Grouped by the Number of Users that Have Used Them. Commands Used By Only One User (Unique Commands) Have Lowest ID's, Commands Used By All Users Highest ID's. Within Each Plot, Horizontal Lines Are Drawn to Separate Groups.

mated from the training data set. The test data is then used to evaluate the test statistic.

Let N_i denote the length of the test data sequence for user i and N_{ic} the number of times command c appears in user i's test data. Then $N_i = \sum_{c \in Test(i)} N_{ic}$.

We now define

$$\text{Uniqueness Score User i} = \frac{1}{N_i} \sum_{c \in Test(i)} W_{ic}\, U_c\, N_{ic} \quad (1)$$

where c indexes the set of (distinct) commands $Test(i)$, the weights W_{ic} are

$$W_{ic} = \begin{cases} 1 & \text{if user } i\text{'s training data contains} \\ & \text{command c} \\ -1 & \text{otherwise} \end{cases}$$

and where the uniqueness index U_c is defined as

$$U_c = \begin{cases} 0 & \text{if } c \text{ used by all users} \\ 1/U & \text{if } c \text{ used by all but one users} \\ \ldots & \\ (U-2)/U & \text{if } c \text{ used by only two users} \\ (U-1)/U & \text{if } c \text{ used by only one users} \\ 1 & \text{if never used by any users} \end{cases} .$$

The quantity U_c is 1 minus the fraction of users that have used command c in the training data. It is the same quantity that has been plotted on the vertical axis in Figures 1 and 2.

For each new command the score in (1) either increases or decreases, pending on whether the associated user has used that command before in the training data or not. The amount of increase/decrease U_c is higher for rare commands than for common commands. Hence a user will tend to score high if he/she uses similar commands to the ones he/she used in the training data. The order in which the commands appear does not matter.

Weighted Uniqueness Scores

Discrimination among users based on (1) works well, but we improve on it for the following reason: Suppose user A uses a rare command only a few times, and user B uses that rare command often. In this case test data from user B with many incidences of that same rare command will be assigned a high score when trained on data of user A. The use of that rare command is indicative of user A and should therefore lead to an increase in the score. However, since user A does not use that command very much relative to other users, the use of that rare command is only moderately indicative and the score increase should not be very large.

We modify (1) by changing the definition of the weights:

$$\text{Uniqueness Score User i} = \frac{1}{N_i} \sum_{c \in Test(i)} W^*_{ic}\, U_c\, N_{ic} \quad (2)$$

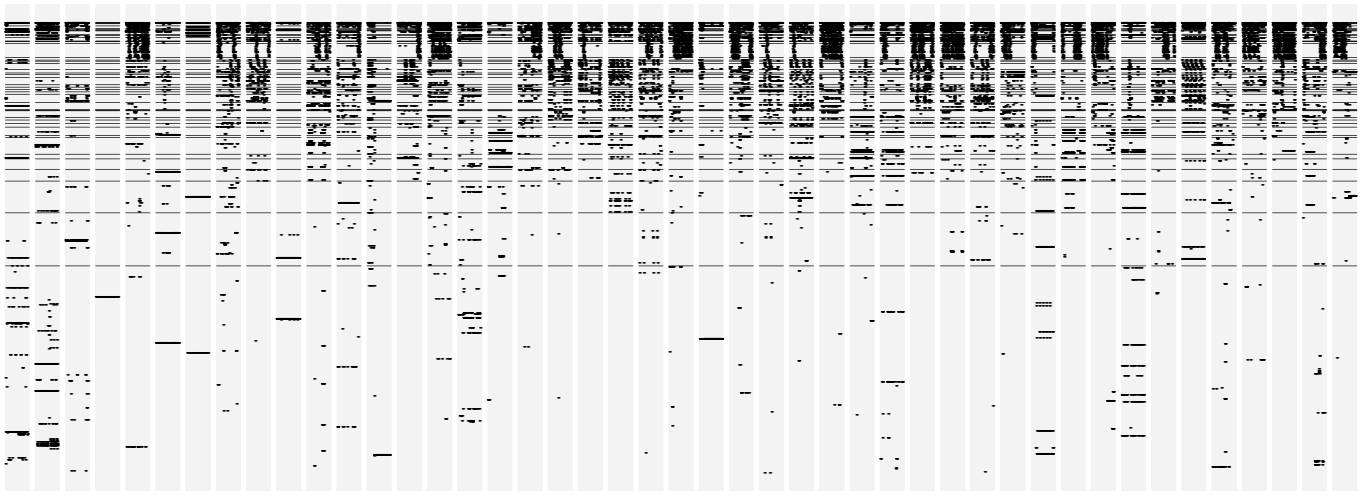

Figure 4: Plots of Command ID vs Command Order in the Audit Stream for Each User. Commands Are Grouped by the Number of Users that Have Used them. Commands Used by Only One User (Unique Commands) Have Lowest ID's, Commands Used by All Users Highest ID's. Within Each plot, Horizontal Lines Are Drawn to Separate Groups.

where U_c is defined as before,

$$W^*_{ic} = \begin{cases} p_{ic}/p_{.c} & \text{if user i's training data contains} \\ & \text{command c} \\ -1 & \text{otherwise} \end{cases}$$

where

$$p_{ic} = N_{ic}/N_{i.} \quad .$$

Commands previously associated with a weight of 1 are now given a smaller weight - how small depends on command usage relative to other users. The weight for a particular command c for user i is high when user i uses command c a lot *relative to other users*, i.e. when the use of command c is especially characteristic of user i. As a consequence, a user using a particular rare command much more often than the legitimate user will score more often for that command when tested as legitimate user, but the impact on the overall score is lower due to a low weight.

4 Results

To evaluate the method presented in the previous section, we compute the test data score (2) of user i ($i = 1, \ldots, 45$) based on the training data for user j ($j = 1, \ldots, 45$) for all pairs of users.

The resulting 45×45 scores are graphically displayed in Figure 5 (only scores greater than zero are shown). Darker shading corresponds to higher scores, lighter shading to lower scores. For the diagonal entries both the test data and the training data correspond to the same user. Ideally, we would be able to perfectly discriminate between diagonal entries and off-diagonal entries, that is all diagonal entries should be darker than all off-diagonal elements.

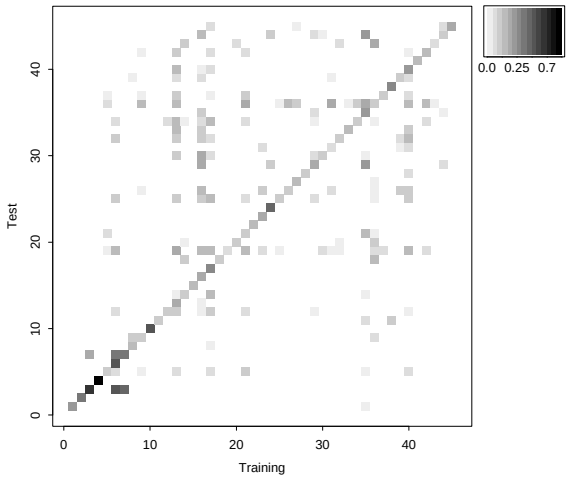

Figure 5: Visual Display of the 45×45 Scores from (2) Based on 1000 Commands. Only Scores Greater than Zero Are Shown. The Darker the Shading, the Higher the Score.

By varying the threshold at which an alarm is raised one can obtain different rates of false alarms (false positives) and missing alarms (false negatives). Figure 6 shows this tradeoff between false positives and false negatives for scores based on 100 commands and 1000 commands. Figure 6 shows two curves in both cases to give an idea of the variability of the curves for different data sets; the second curve in both cases is obtained by interchanging the training and the test data. (The fact that the curves based on 1000 commands do not go all the way to the vertical axis on the left is due to a discreteness effect. The midpoint between zero and one out of 45 possible false alarms corresponds to $1/90 = 1.1\%$ on the horizontal axis.)

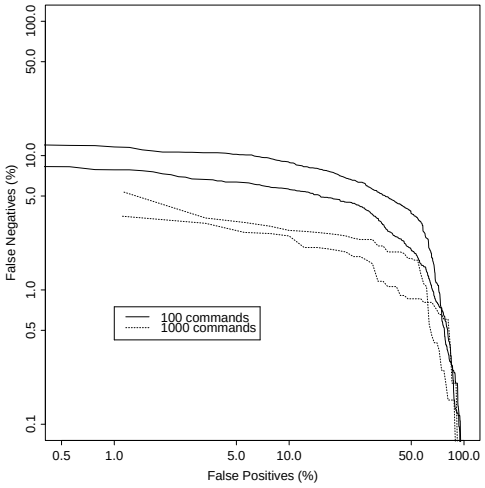

Figure 6: Tradeoff Between False Negatives and False Positives (ROC Curve) Based on (2) For 100 and 1000 Commands. Two Curves Are Obtained by Interchanging the Training and the Test Data. Both Axes Are on a Logarithmic Scale.

The lower left corner on this plot represents the ideal situation: no false alarms, and no missing alarms. For our method no false alarms correspond to about 10% missing alarms based on 100 commands or 5% based on 1000 commands. As expected the discrimination among users based on 1000 commands is easier than based on 100 commands. As the rate of false alarms increases, the rate of missing alarms initially drops only slowly. Incidentally, the threshold of score = 0 corresponds to the situation when no false alarms are generated. Decreasing the threshold below zero leads to an increase of missing alarms without changing the false alarm rate. This implies that based on this data the legitimate user always has a score above zero, whereas most but not all of the other users have a score smaller than zero.

5 Discussion

Intrusion detection based on command uniqueness is conceptionally very simple. It requires very little storage (W_{ij} for all i, j and U_j for all j) and, since (2) is a weighted average, it can be easily updated. Moreover, each update is computationally fast requiring only a few multiplications. We would also like to point out that while (2) constitutes a quantitative improvement over (1), the uniqueness index is qualitatively more important than the weights.

When too many of the alarms generated are false, they tend to be ignored altogether after a while. Therefore, a low false alarm rate is particularly important in intrusion detection. Choosing a threshold of 0, based on 100 commands we are able to avoid false alarms altogether in our experiment. We have not seen another intrusion detection method that has virtually no false alarms at an acceptable rate of missing alarms.

The uniqueness approach may work especially well in the diversity of a research environment. The UNIX operating system allows and even encourages this diversity. The uniqueness approach may not work as well in an environment where everybody's job description is very similar.

The choice of using 100 commands for the scores was arbitrary. We are investigating whether we can discriminate users based on even less commands.

It is difficult to compare various intrusion detection methods due to the lack of a common yardstick. Different methods are based on different data sources and are often too complicated to reimplement for comparison purposes. Based on Figure 6, the uniqueness approach compares favorably with DuMouchel and Schonlau (1998) who report false negative rates between 10% and 50% at 5% false positives for comparable data sources.

Acknowledgements

M. Schonlau's work is funded in part by NSF grants DMS-9700867 and DMS-9208758. We are grateful for feedback from our network intrusion group with members from AT&T Labs Research, The National Institute of Statistical Sciences and Rutgers University. Their comments have led to a number of improvements.

References

Anderson, James P. (1980). Computer security threat monitoring and surveillance. Technical report, James P. Anderson Co., Fort Washington, PA, April 1980.

Computer Immune Systems, (accessed June 23, 1998), `http://www.cs.unm.edu/~forrest/`

COAST project, (accessed June 23, 1998), `http://www.cs.purdue.edu/coast/`

DuMouchel, W. & 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).

Emerald, (accessed June 23, 1998), `http://phlox.csl.sri.com/emerald/`

Intrusion Detection for large networks, (accessed June 23, 1998), `http://seclab.cs.ucdavis.edu/arpa/`

Netranger, (accessed June 23, 1998), `http://www.wheelgroup.com/netrangr/1netrang.html`