Incorporating Soft Computing Techniques into Anomaly Intrusion Detection Systems

Yingbing Yu\(^1\), Han Wu\(^2\)
\(^1\)Department of Computer Science & Information Technology
Austin Peay State University, Clarksville, TN, USA
\(^2\)Department of Mathematics & Statistics
Minnesota State University, Mankato, Minnesota, USA

Abstract - One critical threat of inside attacks facing many organizations is from masqueraders, internal users or external intruders who exploit legitimate user identity and manipulate the system of performing malicious attacks. Intrusion detection systems can be used to build a user profile and a large deviation from the past behavior patterns indicates a possible illegal access from a masquerader. In this paper, we first introduce the typing biometrics of keystroke patterns and apply the probabilistic neural network for the classification. A second behavior profiling approach from command sequences generated by a user is to build the behavior model using the finite automata. New activities of users are compared with the finite automaton for the deviation. A fuzzy inference system is then applied to integrate the two input variables to evaluate the overall threat of the possible masquerader existence. Experiments show promising results with a high detection rate and a low false alarm rate.

Keywords: Anomaly Intrusion Detection, Masquerader Detection, Probabilistic Neural Network, Fuzzy Systems

1 Introduction

Intrusion detection systems (IDSs) attempt to perform the process of monitoring computer networks and systems for violations of security policy (a set of laws, rules, and practices that define the system boundaries) [1]. IDSs can be categorized into two classes based on different detection approaches. Misuse (knowledge or signature-based) IDSs look for specific patterns that define a known attack. The information about known attacks and vulnerabilities of a system is encoded into “signatures”. Any actions that trigger the matches will be reported as “attempts” of intrusions. Anomaly (behavior-based) IDSs assume the deviation of normal activities under attacks and perform abnormal detection compared with predefined system or user behavior reference model.

Anomaly IDSs can be used to build system or user behavior profiles to detect inside attacks from masqueraders, internal users or external intruders who exploit legitimate users identification and password that one may obtain illegally and then perform malicious attacks from the inside. Inside abuse of network access has been cited as the second most cited forms of attacks [3]. To prevent a system from attacks due to identity theft, the effective approach is to monitor user behavior and report any suspicious activities. Alarms are alerted when a user behaves out of characters and a large deviation with the behavior profile is detected.

The goal of this paper is to distinguish a masquerader from genuine users which is a challenging task due to the problem of concept drift, where the observed user behavior may change with different tasks, time, general knowledge level and such other uncertain elements [4]. In this paper, we introduce a soft computing based model to detect masqueraders based on two different user behavior profiling approaches - typing biometric as keystroke patterns and user activities as command sequences generated in a UNIX/Linux system.

The probabilistic neural network (PNN) is applied for the classification of keystroke patterns as normal or abnormal after the biometric template is constructed. Command sequences ordered by date and time are used to build a finite automaton which is identified with the behavior model for a user. A command block of new activities from a user is compared with the finite automata and the deviation is evaluated as memberships with fuzzy sets. A fuzzy inference system is further introduced to integrate the information from the typing biometrics and behavior profile from the command sequence analysis to evaluate the overall threat of a case as the possible masquerader existence in a computing system.

The rest of this paper is organized as follows. Section 2 is the literature review that discusses related research in masquerader detection from command sequences and typing biometrics. Section 3 presents the typing biometrics template for a user and the classification of patterns using the probabilistic neural network. In section 4, we introduce the finite automata based behavior profiling from user
command sequence and the deviation evaluation. Section 5 proposes the fuzzy reasoning system from the input of typing biometrics and behavior profiling of commands to evaluate the degree of overall threat. Experimental results conducted in a data set are given in section 6. The paper concludes with section 7, which discusses the future research work.

2 Literature Review

User behavior profiling can be used for classification, future behavior prediction and masquerader detection. Traditionally user behavior in a system is characterized by parameters such as login frequency, location frequency, last login, session elapsed time, password fails, location fails, amount of network traffic, resources used by user in a session and so on [5]. In this paper, typing biometrics of keystroke patterns and command sequences concatenated by date order are selected to capture a user normal behavior model and detect masqueraders based on the degree of deviations.

De Ruet et al. developed a software methodology that improves security by using typing biometrics to reinforce password authentication mechanisms [2]. Typing biometrics is the analysis of a user's keystroke patterns. Each user has a unique way of using the keyboard to enter a password; for example, each user types the characters that constitute the password at different speeds. The methodology employs fuzzy logic to measure the user's typing biometrics.

Machine learning and statistical methods have been widely used in the literature for the behavior profiling from the analysis of command sequences. Davison and Hirsh developed a model called IPAM (incremental probabilistic action modeling) to predict sequences of user actions [6]. Single-step command transition probability is estimated from training data. Balajinath introduced GBID (Genetic Based Intrusion Detector) to model individual user behavior with a 3-tuple vector which is learnt later by a genetic algorithm [11]. Ryan used a back propagation neural network NNID (Neural Network Intrusion Detector) to identify users simply by what commands and how often they use, called the 'print' of a user [7].

Lane and Brodley [8] selected a machine learning algorithm IBL (instance based learning) to measure the similarity between the most recent 10 commands of a user and the profile extracted from the past. The similarity measure is the count of matches of a new sequence with the sequences from a user’s commands history, with a greater weight assigned to adjacent matches. Schonlau [9] selected several statistics-based methods to detect masqueraders, including uniqueness, Bayes one-step Markov, Compression, Multi-step Markov chain etc. Maxion and Townsend [10] applied Naïve Bayes classification algorithm to user profiling with command-line data, which shows improvement over the best approach of Schonlau [9].

3 Typing Biometrics Classification

Typing biometrics is used for the analysis of a user’s keystroke patterns. Each user types the characters that constitute a command at different speeds. Our method is to build a biometrics template to be used for the anomaly detection. When an authorized user is accessing a system and typing commands, the interval time between two successive characters in a unique command is recorded. For example, if the user types a command “dir”, the time intervals between the characters “d” and “i”, “i” and “r”, “r” and the “ENTER” key will be stored as the biometric characteristic.

If the command contains parameters, the time between the space and a regular character is also recorded. For example, if a user types the command “dir –p”, we will also calculate the time interval between the characters of “r” and the “BLANK” key. A user may type capital letters instead of lower case ones, and this will involve a larger time interval since the user needs to press the “CapsLock” key at the same time. We have not considered this scenario and experiments show that most users prefer to always type commands in lower case. In addition, we only consider those frequently typed commands by a user. The reason is that later a neural network is used for the classification and a relatively large training data set is required for that.

The user typing biometrics in the form of time interval of successive characters in commands will be used as the typing template for the user. On subsequence access to the system, each of the user command typing will go through a neural network system for the classification as normal from the genuine user or abnormal from a masquerader. We have chosen the supervised probabilistic neural network (PNN) for the classification purpose. PNN [12] has the learning and generalization ability of backpropagation multi-layer neural networks (BPNN) and is simpler and faster. It can identify the commonalities in the training examples and then perform classification of unseen examples from the predefined classes.

PNN consists of three feed-forward layers: input layer, pattern layer, and summation layer. An input layer contains as many elements as the number of characters in a command. The pattern layer represents a neural implementation of a version of Bayes classifier and can provide an optimum pattern classifier to minimize the
expected risk of wrongly classifying an object. It gets closer to the true underlying class density functions as the number of training samples increases if the training set is an adequate representation of the class distinctions. The summation layer (output layer) has two elements of normal and abnormal classes to denote if the typed command is from the genuine user or a masquerader. A compete transfer function on the output of the second layer picks the maximum of these probabilities, and produces a “1” for the normal class and a “0” as an abnormal case.

Since a user session usually involved the execution of multiple commands, we pick up a block size of 50 commands for the masquerader detection. To determine if the whole block of commands is from the genuine user or a masquerader, we use a fuzzy inference system for the deviation evaluation from the biometric template. To denote the degree of deviations, three fuzzy sets “low”, “medium” and “high” and fuzzy membership functions are defined to measure the magnitude of these values. The common triangular fuzzy membership is chosen. The common triangular fuzzy membership is chosen in Figure 1. Given an example, if 8 commands in a block of 50 are classified by the NN as abnormal, the degree of this case from a masquerader is low with a degree 0.6 and medium with a degree 0.4.

![Figure 1. Memberships of Deviation for Typing Biometrics](image)

To construct finite automata as the user behavior model, the sequence of user commands concatenated by temporal order is divided into blocks of \( n \) commands in each, called a pattern. From the very beginning of a sequence, each time a pattern is searched and the position moves forward by 1 using the sliding window technique. For a sequence of \( l \) commands, there will be \( (l - n +1) \) patterns. If a pattern is not seen before, a new state is created in the finite automaton to hold it and a connection from previous state to the new one is added. The connection is labeled as the last element of the previous pattern and a number indicating the transition frequency.

New activities of user command sequences are divided into patterns of the same size. Each pattern is compared with the finite automaton to determine the degree of mismatch denoted by numerical values. The sum of mismatch values of all the patterns in a case is regarded as overall mismatch for the test case. Not only the exact mismatch of a pattern with the finite automaton, but also if there are correct transitions between states, will be checked to determine the two values. Three rules have been created for different scenarios.

1. Finite automaton doesn’t have a state associated with a pattern, mismatch=1.
2. Finite automaton has a state associated with a pattern but without a transition from previous state, or the existed transition not correctly labeled, then mismatch=0.5.
3. Finite automaton has a state associated with a pattern and a correctly transition. If the frequency of this transition occurred very few in the finite automaton, then mismatch = 0.2. If the transition occurrence is high, mismatch = 0.

A fuzzy inference system and the corresponding fuzzy membership definition in the Figure 2 are given to evaluate the overall deviation of a case (a sequence of commands in a block) based on the command sequence analysis.

**Rule 1.1**: If the mismatch is low, then the deviation with the profile is low.

**Rule 1.2**: If the mismatch is medium, then the deviation with the profile is medium.

**Rule 1.3**: If the mismatch is high, then the deviation with the profile is high.

4 Profiling from User Commands

To detect masqueraders, we also analyze user activities in the form of commands sequences in addition to the typing biometrics. This approach is more effective in UNIX/Linux system where most of the activities involve the shell commands execution though it can also extend to the Windows systems. We first present the finite automata model to capture the behavior patterns from the sequential data of commands. Then we describe how to measure the mismatch with the profile and evaluate the degree of deviation using fuzzy logic.
5 Fuzzy System Threat Evaluation

The information from the user typing biometrics and command sequence analysis is integrated for the analysis since this will provide more accurate judgment about new behavior patterns. We create a fuzzy inference system based on the two factors to evaluation the overall threat as the possibility of masquerader existence.

The variable “biometric deviation” is selected as one input to denote the degree of deviation from the biometric template discussed in section 3. The variable “profiling deviation” is used as another input to represent the mismatch with the established profile from command sequences. For the output, we have chosen five different classes “very low”, “low”, “medium”, “high”, and “very high” to denote the degree of the threat belonging to each fuzzy set. The membership functions are defined in Figure 3 using the triangular memberships.

Since each of the two input variables can be in one of three fuzzy sets “low”, “medium” and “high”, the number of possible fuzzy rules is very large and only those mostly effective ones in our experimental trials are selected for the fuzzy inference system.

Rule 2.1: If the biometric deviation is high, and the profiling deviation is high, then the possibility of masquerader is very high.

Rule 2.2: If the biometric deviation is high, and the profiling deviation is medium, then the possibility of masquerader is high.

Rule 2.3: If the biometric deviation is medium, and the profiling deviation is high, then the possibility of masquerader is high.

Rule 2.4: If the biometric deviation is medium, and the profiling deviation is medium, then the possibility of masquerader is medium.

Rule 2.5: If the biometric deviation is medium, and the profiling deviation is low, then the possibility of masquerader is low.

Rule 2.6: If the biometric deviation is low, and the profiling deviation is medium, then the possibility of masquerader is low.

Rule 2.7: If the biometric deviation is low, and the profiling deviation is low, then the possibility of masquerader is very low.

After the membership values of facts with respect to each antecedent in a rule are determined, the MAX-MIN method is applied to measure the impact of fuzzy rules and the highest membership is selected. For each fuzzy rule, the output membership is obtained using the MIN implication operator to select the minimum membership value of antecedents in the premise. If several fuzzy rules generate different membership values associated with the same fuzzy set, the MAX implication operator is used to select the largest one as the final result.

The center of area (COA) defuzzification method is applied to get a single non-fuzzy crisp output of final threat ($\Delta Threat$) to measure the degree of the deviation with the behavior profile. The COA formula is:

$$\Delta Threat = \frac{\sum_{k=1}^{n} \mu_k \cdot center(k)}{\sum_{k=1}^{n} \mu_k}$$  (1)

where $n$ is the number of fired rules, $\mu_k$ is the degree of membership of rule $k$, and $\text{center}(k)$ is the peak-value where the fuzzy set for the rule $k$ has the maximum membership values. This final $\Delta Threat$ value can be
compared with a threshold value to determine the degree of the threat to the system. In our experiments, if the overall $\Delta$Threat is great than 0.6, a test case will be classified as abnormal from a masquerader. Otherwise the case will be regarded as “normal” that the genuine use is accessing the computing system.

6 Experimental Results

The proposed model based on soft computing techniques is applied to detect masqueraders using the data generated in a network laboratory. During that time, we have collected user commands of about 35 students who worked on Linux systems over a period of about three months. Normal data was generated when they worked on several programming and lab assignments. The masquerader data was generated when the students tried to compromise computers in a local area network, as part of lab assignments.

The data generated in the first two months in the laboratory is used to build the typing biometrics and behavior profile for each user. The data generated in the third months when some of the users conducted attacks on the network is served as the masquerader cases. Since some user didn’t generate a large amount of data and only 25 users are selected as intrusion targets and the normal training data set is 2148 command blocks of 50 commands in each. The number of normal test cases is 436 blocks and the number of anomaly cases from masqueraders is 545.

For the 545 cases of masquerader data, 508 are classified correctly as “abnormal” and only 37 cases are mistakenly identified as “normal”. For most users, the model can achieve a high masquerader detection rate between 90% and 95%. The average detection rate is about 93.2% (508/545) and the missing rate is 6.8% (37/545). We also tested the 436 normal cases to check if the model can successfully do the classification. If a normal one is classified incorrectly as abnormal, a false alarm was generated. In total, only 18 normal cases are incorrectly identified as “abnormal” and the false alarm rate is 4.1%.

In addition, we have conducted further experiments with other different block sizes (# of commands in a block) to compare the performance of detection and false alarm rate. For a real intrusion detection system, it is the ultimate goal to detect masqueraders within a short time interval and alert the system earlier to prevent further information loss. Based on the results, we have found that the interval of about 50 commands execution achieves both a high detection rate and lower false alarm rate. When a larger test block size is selected, the model can still achieve a high detection rate but will introduce a higher false alarm rate.

If the block size is above a threshold (e.g., 100 commands), the model has a low detection rate and high false alarms. The possible reason is that the probability of overlapping behavior patterns increases rapidly when target size is above a certain threshold. In practice, an appropriate size can be selected based upon specific security policies for an organization. In general, it is fairly reasonable and effective for an anomaly IDS to detect potential masqueraders after just about 50 commands execution.

7 Conclusions

In this paper, we introduce the soft computing techniques in the area of intrusion detection systems to detect masqueraders based on two different approaches of user behavior profiling. Typing biometrics of keystroke patterns from users can be collected in the training phase as the biometric templates. Another behavior profiling approach from user activities of commands execution is also applied where the sequential data is to build a finite automaton. To integrate the valuable information from both the approaches, a fuzzy inference system is presented based on the two inputs – deviations by comparing with the biometric template and finite automaton.

We also want to extend the current research of masquerader detection. As we have notices that in the last decade GUI and Internet-based applications have been deployed in both UNIX and Windows systems. A large part of user activities associated with these applications may not involve individual commands directly entered into the system, but instead consist of mouse clicks on icons. The behavior modes from this kind of activity will differ significantly from those discussed in this paper. Future work would address these questions.

8 References


