A Survey on Masquerader Detection Approaches

Maximiliano Bertacchini[1] and Pablo I. Fierens[2]

[1] CITEFA - Instituto de Investigaciones Científicas y Técnicas para la Defensa
San J. B. de La Salle 4397 (B1603ALO),Villa Martelli, Buenos Aires, Argentina
Tel.: 5411-4709-8285, Fax: 5411-4709-5363
mbertacchini@citefa.gov.ar
[2] ITBA - Instituto Tecnológico Buenos Aires
Av. Eduardo Madero 399 (C1106ACD), Buenos Aires, Argentina
Tel.: (5411) 6393-4822
pfierens@itba.edu.ar

Abstract. This paper presents a survey on the area of masquerader detection. The three most popular publicly available UNIX command-line datasets are showed and their features are compared. Several different masquerader detection approaches are reviewed and their results are compared applying the most popular measures of detection effectiveness in this area, introducing the most extensive quantitative comparison of results in literature. Possible ways for future work in this area are proposed as well.

Keywords. masquerader detection, command-line behavior, computer security, comparison, review, survey.

1 Introduction

In computer security, a masquerader is an intruder trying to impersonate a legitimate user. Early and effective intrusion detection is a critical factor in securing a computer system.

We are focusing on the detection of masqueraders in the UNIX command line history of computer users. Despite the prevalence of graphical user interfaces (GUIs), user interfaces based on a command line are still popular. Moreover, the same techniques can be applied for anomaly detection on other sequential data, such as sequences of system calls at the operating system level, packet flow in network activity, and even in other areas such as genetics and bioinformatics.

In the last decade, a vast amount of work has been published on this subject, with approaches covering a wide range of techniques such as statistics, artificial intelligence, data mining and bioinformatics. However, most of them share many details in common, such as the used dataset, data configuration and general training and testing methodology. Masquerader detection approaches have shown a rather steady progress with time, and probably better techniques will be conceived in the future.

Our goal is to present the most representative works to our knowledge in this area, from the seminal work by Schonlau et al. [1] in 2001, on to the present. This survey could serve as an introduction to the literature on the subject of masquerader detection for new researchers. We are also interested in exposing topics in this area which could be subject to further research.

This work is structured as follows: Section 2 presents the most popular publicly available UNIX command-line datasets and some of ther shortcomings; Section 3 shows a review of masquerader detection approaches in literature; Section 4 briefly describes the most popular measures for evaluation of detection results; Section 5 makes a quantitative comparison of results achieved by the reviewed techniques; Section 6 presents our conclusions and possible areas for future work.

2 Datasets

2.1 Schonlau et al.

Matthias Schonlau et al. introduced their own dataset in [1] and [2] in 2000. It is also known as SEA dataset (Schonlau Et Al.), and it is freely available for download at Matthias Schonlau's Internet homepage[3]. This dataset has since become the de-facto standard dataset and is widely used in the majority of papers about masquerader detection. Its associated configuration is also the most widely used. This allows, in principle, the comparison of results among different detection approaches.

This data set consists of commands collected from UNIX `acct` audit data. Of all fields of audit data provided by `acct` only the username and the command were taken. Data contains 50 different users with 15000 commands each. The first 5000 commands are considered genuine. The remaining 10000 commands of each user are divided into 100 blocks of 100 commands each. These blocks are seeded with masquerading users, i.e. with data of another user not among the 50 users. A block is a masquerader with a probability of 1%. If a block is a masquerader, then the following one is a masquerader too with a probability of 80%. As a result, approximately 5% of the test data contain masquerades. Not all users' testing data have masquerades interspersed.

Commands have no arguments nor any parameters, as this additional information was not provided by `acct`. Due to the way `acct` collects audit data from the system, it is impossible to tell commands typed by human users on the command line prompt from those run automatically from shell scripts. This can lead to some users having a very regular pattern of few commands, due to the repetitive automated execution of shell scripts.

1v49 Configuration An alternative configuration to the SEA dataset was proposed by Maxion et al. in [3] to address some methodological shortcomings found

[3] Matthias Schonlau's homepage: `http://www.schonlau.net`.

in the original configuration, such as the fact that different masqueraders were injected into different users, and some users didn't even get any masquerader. This makes evaluation and error analysis very difficult. In 1v49 configuration, for each user the first 5000 commands of the other 49 users are used as masquerader data for testing purposes. Despite its methodological advantages, this configuration has not been widely used. Note that this configuration does not simulate the presence of masqueraders which is expected in real world data.

SEA-I A variation on the original SEA dataset is proposed by Posadas et al. in [4], called SEA-I, where the masquerader blocks are replaced by synthetic blocks created taking into account the command frequency of each user, triying to emulate the behavior of an intruder who tries to act like the legitimate user. As a result, masquerader detection on this dataset is more difficult, and more complex techniques are needed[4].

2.2 Greenberg

This data set was collected by Saul Greenberg as part of his PhD thesis in 1988 [5]. It contains data from 168 Unix users using *csh* (C shell) as command line shell. Greenberg classified users in four groups as follows: Novice Programmers, Experienced Programmers, Computer Scientists and Non-programmers. This dataset is available at Saul Greenberg's homepage[5].

Data was collected and stored in plain text files which contain the following information: session start time, session end time, the command line as entered by the user, the current working directory, the alias expansion of the previous command (if any), an indication whether the line entered has a history expansion or not, and the error detected in the command line (if any).

This dataset was first used for masquerader detection purposes by Maxion et al. in [6]. The main advantage of this dataset in contrast with previous ones such as SEA is the availability of additional information for each command which may help to improve the effectiveness of masquerader detection. However, in practice, of all these additional data, only the arguments, parameters, and alias are used.

This dataset is often used in two different configurations simultaneously: *enriched* and *plain*. The enriched configuration includes the command, alias, arguments and parameters. The plain configuration only uses the command and alias, thus effectively mimicking the configuration of Schonlau et al's dataset. This allows the comparison between enriched and plain configurations of the same dataset in order to verify the hypothesis that these additional data improve the detection effectiveness.

2.3 Purdue University

Terran Lane and Carla E. Brodley introduced a dataset in 1997, which was later known as Purdue dataset or just "PU" dataset [7]. It consists of the UNIX shell

[4] The authors of [4] present an alternative which is discussed in Section 3.3
[5] Saul Greenberg's homepage: http://pages.cpsc.ucalgary.ca/~saul/

command histories of 4 users of the Purdue Millenium Lab, collected during a four month period. It is available at the UCI KDD archive[6]. The number of collected commands per user goes from 7769 to 22530, with an average of approximately 16500 commands per user. Commands are enriched, that is, command names, arguments and options are preserved, but filenames are ommited, due to the intuition that the *behavior* of users is more significant for their profiling than *content*. Very few works on masquerader detection use this dataset, probably because of the low number of users it contains.

2.4 Synthetic Datasets

Synthetic datasets provide a non-intrusive alternative to traditional ones, with the benefit of rapid availability and high customizability.

In 2004 Chinchani et al. introduced a framework for template-based command line data generation called RACOON [8]. It produces sequences of commands based on templates, which can be specified manually or created from any other data set. A template is composed of the following components and their statistical properties: commands, meta-commands, jobs, meta-command sequences, job hops and sessions. A session is basically a sequence of jobs, which are sequences of meta-commands. A meta-command is a set of similar commands in terms of their function.

One limitation of RACOON is that it is able to generate only truncated commands, not enriched ones. On the other hand, it can provide large amounts of command-line data, which is otherwise very difficult to obtain.

2.5 Comparison of Datasets

Table 1 shows a comparison between the previous three UNIX command-line datasets. As shown above, all of them consist of data from legitimate users. No public dataset of real intruders is known to the authors of this paper. Furthermore, although a thorough analysis and modelling of the behavior of intruders is lacking in this area, it is generally assumed that masqueraders can be detected by applying any of the techniques reviewed below. The number of users varies from just a few to more than one hundred; the size of each user's dataset ranges from hundreds to tens of thousands of commands; and commands may be truncated. These datasets are also more than ten years old. Hence, a new UNIX command-line dataset improving on these features would be extremely useful. It could include real intruder data collected from honeypots and honeynets, should be sufficiently large, and should include additional pieces of information such as command parameters and options, and login sessions.

3 Review of Masquerader Detection Approaches

Masquerader detection approaches can be classified by means of several different criteria. Independently of the applied detection technique, most approaches are

[6] The UCI KDD archive: http://kdd.ics.uci.edu

Table 1. Comparison of publicly available UNIX command line datasets

Dataset	Users	Cmds (total)	Cmds/user (average)	Enriched	Contaminated	Sessions	Year	Time lapse
Greenberg	168	303628	1807	Yes	No	Yes	1988	4 months
PU	4	66177	16500	Yes	No	Yes	1997	4 months
SEA	50	750000	15000	No	Yes	No	2000	several months

based in a scoring system, where each block gets a score and is classified as normal or masquerader according to some detection *threshold*. We have chosen to subdivide these techniques according to the classifier algorithm each of them is based on. Other distictive criteria for the classification of masquerader detection methods could be: data preprocessing and user profile construction, user profile updating, size of testing blocks (number of commands), the used dataset and its configuration.

3.1 Information-Theoretic-based Approaches

Schonlau et al. first proposed a rather simple compression-based approach in [1], called the Compression method. It is based on the premise that data from the same user compresses more readily than mixed data from different users. The score of a testing block is defined as the number of additional bytes needed to compress it when appended to the training data. The UNIX command `compress` was applied, which is based on the popular Lempel-Ziv algorithm [9]. In [1], results from the Compression method seem discouraging in comparison with the other five methods presented there.

In 2006 Bertacchini and Fierens applied a more sophisticated compression-based approach, the Normalized Compression Distance (NCD) on the SEA dataset [10]. NCD is a method for clustering and classification developed by Vitányi et al. [11], which is based on the notion of Kolmogorov Complexity [12]. This function is not computable, but may be approximated using real-world compressors such as `gzip` or `bzip`. Therefore, NCD relies on external real-world compressors to calculate the difference or "distance" between any two objects. In this case, the two objects being a suspicious block of commands and a user's profile or their command-line history. Results were better than the Compression approach, and similar to Schonlau's best results. A later work by the same authors [13] showed improved detection applying the NCD on Greenberg's enriched command-line data.

In 2007 Evans et al. proposed a grammar-inference algorithm called MDL-compress [14] which uses Minimum Description Length (MDL) principles from the theory of Kolmogorov Complexity and Algorithmic Information Theory [12] to model legitimate user activity and use the resulting grammar to detect masqueraders. This algorithm is reported to produce results very similar to those of Schonlau et al.'s Compression approach [1], though no quantitative results are available to our knowledge.

3.2 Text Mining-based Approaches

Latendresse proposed in 2005 a text-mining approach [15] based on the Sequitur algorithm. The Sequitur algorithm extracts hierarchical structures from a string by constructing a context-free grammar [16]. For each user, a Sequitur grammar is generated to extract the repetitive sequences of commands and their associated frequencies, and each testing block is compared recursively with the legitimate user's profile. This method achieved high detection with low false alarm rates.

In 2004, Oka et al. proposed a masquerader detection method based on co-occurrence matrices in [17], called Eigen Co-occurrence Matrix (ECM). The ECM method models a user's behavior by correlating an event in a sequence with the following events appearing within a certain distance. It extracts the principal features of the so-called co-occurrence matrix of a user's sequence of UNIX commands through Principal Component Analysis (PCA) [18], thus reducing its inherent high dimensionality, and obtains a feature vector which can be classified with any classification technique based on vector representation. In this case, a simple Euclidean distance measure was used, achieving relatively poor detection results.

Oka et al. tried to improve their previous results in [19]. This approach builds a layered network to define the user's profile by multiplying the feature vector by the Eigen co-occurrence matrix. The same procedure is applied on each testing sequence and the resulting layered network is compared with the profile of the corresponding user; it is classified as anomalous or normal based on a per-user threshold. The results were promising, achieving a detection rate and false detection rate similar to those of the Naïve Bayes method. This approach has high computational cost, particularly during its training phase.

In 2006 Chen and Dong introduced a method called "One-Class Classification using Length Statistics of Emerging Patterns" (OCLEP) [20] and compared its detection effectiveness with OCSVM [21] on the SEA and 1v49 datasets. Emerging Patterns (EPs) are patterns whose support increases from one user command history to another with a big ratio. For classification, the lenght statistics of EPs extracted from data are used. Obtained results are similar to those of OCSVM [21], though with slightly higher false alarm rates.

3.3 HMM-based Approaches

A Hidden Markov Model (HMM) is a statistical model where the modeled system is assumed to be a Markov process with unobserved state. An HMM is defined in [22] as a doubly stochastic process with an underlying stochastic process that is not observable (it is hidden), but can only be observed through another set of stochastic processes that produce the sequence of observed symbols.

R. Posadas et al. [4] proposed a hybrid approach using session folding and Hidden Markov Models (HMM). It introduced a variation on the SEA dataset, called SEA-I (see Section 2). It applies the `Sequitur` algorithm [16] to extract the grammar from every session to compress them into production rules, which in turn are used as training data for the HMM model. The grammar extraction

process is called *session folding*. The detection results of this method are similar to those of the Uniqueness method [1] on the SEA dataset, but are better on the SEA-I dataset, where it outperforms the other three tested methods, showing its robustness against masquerader attempts based on command frequency due to its grammar extraction features.

An immunity-based approach for masquerader detection was presented in [23] and [24], and later extended in [25]. It consists on multiple agents, each trained to recognize the profile of one user, in a similar way to how organic immunity systems work by distinguishing "self" cells and tissues from "nonself" ones. A Hidden Markov Model is used to construct user profiles, but the approach is independent of the profile construction method. Each testing block gets a score from every agent and, by means of a voting mechanism and a threshold value, the legitimacy of that block is decided. Results were slightly better than those of Uniqueness [1].

3.4 Naïve Bayes-based Approaches

The Naïve Bayes classifier is one of the most popular classifiers in text classification. They are simple, probabilistic classifiers known for their inherent robustness to noise and their fast learning curve. It is generally used with the "bag of words" model, which profiles documents based on word frequencies, ignoring the sequence information [26].

The Naïve Bayes classifier was first applied on Schonlau's dataset by Roy A. Maxion and Tahlia N. Townsend in 2002 [3] where, additionally, the alternative 1v49 dataset configuration was proposed. The Naïves Bayes approach with online profile updating, using the bag of words model, performed remarkably well and achieved one of the best detection results known to the authors of this paper. This paper was later extended by the same authors in [27] including a thorough error analysis, and in [28] by investigating a possible flaw in the original Naïve Bayes approach which prevented the detection of so-called "super-masqueraders". Jia and Maxion later found that users who are more successful in Naïve Bayes self-recognition can achieve better results in masquerader detection [29]. Recently, Killourhy and Maxion presented a "fortified" version of the Naïve Bayes method which is able to detect super-masqueraders by adding a detector of never-before-seen commands (NBSCs) [30]. Maxion et al. extended their previous work by applying the Naïves Bayes classifier on Greenberg's enriched command line data [6]. Furthermore, the block size was reduced from 100 commands to 10 commands, showing that real-time masquerader detection could certainly be possible to achieve.

Yung further extended the original work by Maxion et al. by trying a user profile updating scheme with feedback [31], in which the user is asked for confirmation about the classification of command blocks. The user feedback scheme showed improved results in comparison with those with no feedback, showing that user feedback is a valuable input to any masquerader detection algorithm.

Yung [32] presented an extension of the Naïve Bayes classifier used in [3] called self-consistent Naïve Bayes, which iteratively updates the user profile on

each new block, based on the expectation-maximization algorithm [33]. This technique reduced the missing alarm rate by 40%.

3.5 Sequence-based and Bioinformatics Approaches

The sequential nature of UNIX command-line data lends itself to many different approaches from diverse disciplines. Techniques used in bioinformatics have been applied with promising results. Sequence alignment is a tool commonly used to quantify the similarity between two genetic sequences.

The Sequence Match approach tried by Schonlau et al. in [1] is based on previous work on masquerader detection by T. Lane and C. E. Brodley [7]. It computes a similarity measure between the ten most recent commands and a user's profile by counting the number of matches in a command-by-command fashion. This method showed good detection for very low false alarm rates (below 1%), but otherwise it did not achieve good results.

Gebski and Wong [34] modeled user sessions as trees where each node represents a command and the edges represent the probabilities between them. This approach was tested on the PU dataset, with good detection rates at the cost of high alarm rates.

Coull et al. [35] used a variation of the Smith-Waterman algorithm [36] for sequence local alignment to compute a semi-global alignment similarity between a user profile and a suspicious block of commands. Their scoring algorithm penalizes the insertion of gaps both into the user signature and the testing block and, of course, rewards matching commands. This method achieved a good hit rate, similar to that of Naïve Bayes, but with a slightly higher false alarm rate. Coull and Szymanski have recently improved this algorithm using a signature updating scheme with binary scoring, achieving good detection rates with a low false alarm rate [37].

3.6 SVM-based Approaches

Support Vector Machines (SVMs) are a set of machine learning algorithms used for classification and regression. SVM classifies data by constructing a separating hyperplane in the n-dimensional feature space of training inputs, which maximizes the margin between them [38].

Wang and Stolfo's work [21] introduced the application of one-class training for masquerader detection. Their work compared the use of the Naïve Bayes classifier with SVM, in both one-class and multi-class variants, on the SEA dataset. They also compared two common models of data representation: multivariate Bernoulli, or binary; and multinomial model, also known as "bag of words". The best results were obtained by one-class SVM using binary features and two-class multinomial Naïve Bayes, achieving similar hit and false positive rates as those previously obtained by Schonlau et al. [1].

Chen and Aritsugi [39] tried to reduce the computational cost of the ECM method [19] by rearranging the co-occurrence matrices (as they are generally

sparse), which they called "enhanced Co-occurrence Matrix", and then applied one- and two-class SVM with updating for classification. Results were similar to previous studies, though false alarms were slightly higher. Two-class SVM produced better results than one-class SVM.

In 2006, Li, Li and Liu [40] also tried to reduce the computational burden of the ECM method [19], as well as reduce overfitting, by extracting the principal features of user behavior from the correlation eigen matrix using Principal Component Analysis (PCA) [18], and feeding these attributes to an SVM classifier, on the SEA dataset. Results are among the best reported so far, but very few details are available about this approach.

Kim and Cha applied SVM with a voting engine on SEA, 1v49 and Greenberg datasets [41]. In the voting scheme, each block of commands is divided into overlapping subblocks which are tested for masquerader activity and a majority voting algorithm decides if the whole block is legitimate or not, according to some threshold. Support Vector Machine (SVM) is used for classification, using a radial based function (RBF) kernel. The most common commands are grouped together (i.e., those used by more than a certain number of users, exceeding a certain percentage) and treated as a single command, which produces a lower false alarm rate than that obtained with no grouping of commands. Results show a slight improvement over Naïve Bayes. Unfortunately, it is not clear how the SVM classifier was trained.

Yang, Zhang and Cai applied one-class SVM with a string kernel on both SEA and PU datasets [42]. The string kernel has a practical advantage in that it can directly process the UNIX command sequences. Few details are given about this work. Results show good detection rates, but very high false positive rates.

3.7 Other Approaches

The Uniqueness approach was first introduced by Schonlau and Martin in [2] and was later compared with other five techniques in [1]. A test statistic is defined on terms of unpopular (ie, used only by a few users) and unique (i.e. used only by one user) commands. Only command frequencies are used; the order in which commands appear is not taken into account. This technique achieved the lower false alarm rate among all of Schonlau et al's approaches, but at the cost of a low hit rate.

Ming Dong Wan et al. [43], on the other hand, showed that high frequency commands work as well as unique commands

The Bayes One-Step Markov [1] is based on one-step transitions from one command to the next, using a Bayes factor statistic to test the null hypothesis that the observed one-step command transition probabilities are consistent with the historical transition matrix. This approach, using profile updating, achieved relatively poor results.

Schonlau et al.'s Hybrid Multistep Markov method [1] is based on a high-order, or multilevel, Markov chain. In order to limit the high dimensionality of the Markov model, the number of commands is reduced by grouping together the least used commands and treating them as one command, and a mixture

transition distribution approach is used to model the transition probabilities. This approach showed poor results.

Schonlau et al. applied Incremental Probabilistic Action Modeling (IPAM) [1] on the SEA dataset. This algorithm is based on the one-step command transition probabilities, which are updated continually using an exponential updating scheme. A command is labeled as suspicious if it is not one of the four most probable commands to follow the previous one, according to its transition matrix. Few details are given about this technique. Its results were not particularly promising.

In 2004 Wang et al. introduced an algorithm called Non-negative Matrix Factorization (NMF) [44], which produced good detection results. Mex-Perera et al. later improved [45] the original NMF approach by using a data normalization phase which takes into account the least used commands, assigning them a higher weight.

In 2007 J. Beauquier and Y. Hu [46] tried an intrusion detection model based on the combination of known methods, called Pearson Correlation Coefficients-Rank (PCC-R) Combination on the SEA dataset. They use a notion of distance on different methods, and combine them by applying the Pearson correlation coefficients. The combined methods are: Uniqueness [1], Bayes one-step Markov [1], Naïve Bayes and Probabilistic Finite State Automata (PFSA). For each user, two types of detection thresholds are defined: one for each method and one for their combination. This approach showed improved hit rates (ie, lower missing alarms) compared with previous works, though the false alarm rates were not improved.

4 Evaluation

We review the most popular measures for the evaluation of detection effectiveness. This permits the comparison of different masquerader detection algorithms. Many other measures have been applied in literature, but not as frequently as these, which in most cases makes the comparison of results very hard.

4.1 Misses and False Alarms

A *miss*, *missing alarm* or false negative happens when a masquerader block is classified as legitimate. A *false alarm* or false positive occurs when the system regards a legitimate user as a masquerader. The percentage of missing alarms, or *Missing Alarm Rate*, and the percentage of false alarms, or *False Alarm Rate* (FAR) are the most popular measures of effectivenes in masquerader detection. It is generally accepted that a low false alarm rate is desirable. Schonlau et al. [1] proposed a target false alarm rate of 1%. Sometimes the Hit Rate is used instead of the Missing Alarm Rate. The Hit Rate and the Missing Alarm Rate are complementary percentages, so that $Hits = 100 - Misses$.

4.2 ROC Curves

Receiver operating characteristic curves, or ROC curves, are often utilized to show the trade-off between the missing alarm rate and the false alarm rate, by varying the masquerader detection threshold. They can be plotted in different ways such as Missing Alarm Rates vs. False Alarm rates or Hits vs. False Alarm Rates. ROC curves permit the visual comparison of different techniques applied on the same dataset. The area under the ROC curve (AUC) is a measure to quantitatively compare different ROC curves which, surprisingly, is seldom applied on masquerader detection literature.

4.3 Cost

Maxion et al. proposed a cost function [3] to assess the effectiveness of an intrusion detection scheme in terms of a linear combination of missing alarms and false alarms. False alarms are prioritized as there is a general agreement that a false alarm should be more expensive than a missing alarm. Of course, the lower the Cost function, the better the effectiveness of an algorithm.

$$Cost = Misses + 6 \times FalseAlarms \tag{1}$$

5 Detection Effectiveness Comparison

Table 5 shows a comparison of results achieved by the reviewed masquerader detection approaches. Results are expressed in terms of Hits, False Alarm Rates (FAR), and Maxion et al.'s Cost function (1). Methods are subdivided according to the dataset they were applied on, as a direct comparison is not possible between them. Results are showed in increasing order according to their Cost function, as it is capable to combine Hits and Misses in one result.

Overall, the most effective approaches seem to be those based on SVM or Naïve Bayes. The combination of different methods (as in [46]) also achieved very good results. On the other hand, the use of enriched data shows significant improvements in all cases. This can be clearly seen in the Greenberg dataset subsection in Table 5. Also, methods which were tested with and without user profile updating show an improvement when applying updating.

6 Conclusions

We have reviewed and compared different aspects of masquerader detection, including: datasets, classification algorithms, effectiveness measures and detection results. Their diversity makes comparisons hard or even impossible to achieve. A unified and coherent framework for testing and comparing different algorithms in a consistent, reliable and reproducible way could benefit the entire comunity

[7] Results manually extracted from published graphs

Table 2. Comparison of detection results achieved by different approaches

SEA dataset [1]	**Hits**	**FAR**	**Cost**(1)	**Config.**	**Upd.**
NCD (bzip) [10]	30.4	0.6	73.1	SEA	No
Compression [1]	34.2	5.0	95.8	SEA	Yes
Customized Grammars [15] [7]	65.0	1.0	41.0	SEA	Yes
ECM with Layered Networks [19]	72.3	2.5	42.7	SEA	No
OCLEP [20]	72.59	4.9	56.81	SEA	No
OCLEP [20]	46.63	5.14	84.21	1v49	No
ECM [17]	37	4.5	90.0	SEA	Yes
Immunity-based HMM [23] [7]	60.0	1.0	46.0	SEA	No
Hybrid Grammar HMM (SEA) [4] [7]	45.0	2.0	67.0	SEA	No
Hybrid Grammar HMM (SEA-I) [4] [7]	25.0	3.0	93.0	SEA-I	No
Self-consistent Naïve Bayes [32] [7]	79.0	2.0	33.0	SEA	Yes
Naïve Bayes updating w/ feedback [31] [7]	76.0	2.0	36.0	SEA	Yes
Naïve Bayes [21] [7]	70.0	2.0	42.0	SEA	No
Naïve Bayes [3]	61.5	1.3	46.3	SEA	Yes
Naïve Bayes [3]	66.2	4.6	61.4	SEA	No
Naïve Bayes [3]	62.8	4.6	64.8	1v49	Yes
Seq. Alignment (upd. binary scoring) [37]	68.6	1.9	42.8	SEA	Yes
Seq. Alignment [35]	75.8	7.7	70.4	SEA	Yes
Sequence-Matching [1]	36.8	3.7	85.4	SEA	Yes
SVM [41]	94.8	0	5.2	1v49	No
CEM-SVM with PCA [40]	82.6	3.0	35.4	SEA	No
SVM+ECM two-class [39]	72.24	3.00	45.7	SEA	Yes
OCSVM with String Kernel [42]	62.0	1.5	47	SEA	No
OCSVM [21] [7]	70.0	4.0	54	SEA	No
SVM+ECM one-class [39]	62.77	6.00	73.2	SEA	Yes
SVM [41]	80.1	9.7	78.1	SEA	No
PCC-R combination [46]	78.9	3.2	40.3	SEA	Yes
Improved NMF [45]	42.0	1.0	48.0	SEA	No
PCC-R combination [46]	81.3	4.9	48.1	SEA	No
NMF [44]	61.1	3.9	62.3	SEA	No
Uniqueness [1]	39.4	1.4	69.0	SEA	Yes
Hybrid multistep Markov [1]	49.3	3.2	69.9	SEA	Yes
Bayes 1-Step Markov [1]	69.3	6.7	70.9	SEA	Yes
IPAM [1]	41.1	2.7	75.1	SEA	Yes
High Freq. Commands (1^{st} order) [43]	70.9	9.20	84.2	SEA	No
High Freq. Commands (1^{st} order) [43]	71.0	13.4	109.4	SEA	Yes
GREENBERG dataset [5]	**Hits**	**FAR**	**Cost**(1)	**Config.**	**Upd.**
NCD(bzip) [13]	60.0	1.0	46.0	enriched	No
SVM [41]	87.3	6.4	51.1	enriched	No
Naïve Bayes [6]	82.1	5.7	52.1	enriched	Yes
Naïve Bayes [6]	70.9	4.7	57.3	truncated	Yes
SVM [41]	71.1	6.0	64.9	truncated	No
NCD(bzip) [13]	38.0	1.0	68.0	truncated	No
PU dataset [7]	**Hits**	**FAR**	**Cost**(1)	**Config.**	**Upd.**
OCSVM with String Kernel [42]	60.0	2.0	52.0	-	No
Tree-based Model [34]	85.0	10.0	75.0	-	No

of intruder detection research. This paper tried to show the most complete comparison of masquerader detection methods to date, which we believe may be a first step in that direction.

Though often mentioned as a possible way for future work in literature, little research has been performed on the combination of different masquerader detection techniques and even different datasets, such as key-press timings for keystroke dynamics analysis, network activity logs for event correlation or sequences of system calls, etc.

On the other hand, the lack of real masquerader datasets prevents a thorough analysis on the behavior of intruders and forces researchers to make assumptions about their ways of attacking. A publicly available dataset including these features could be valuable for the improvement of intrusion detection techniques.

7 Acknowledgements

This work is part of a more extendend research project which is supported in part by ANPCyT (National Agency of Scientific Promotion and Technology) under the grant PICTO CITEFA 2004 18623. We gratefully acknowledge financial support from ANPCyT. We would also like to thank all members of CITEFA's Si6 Lab and the peer reviewers for their advice and support.

References

1. Schonlau, M., Dumouchel, W., hua Ju, W., Karr, A.F., Theus, M., Vardi, Y.: Computer intrusion: detecting masquerades. Statistical Science **16** (2001) 58–74
2. Schonlau, M., Theus, M.: Detecting masquerades in intrusion detection based on unpopular commands. Inf. Process. Lett. **76**(1-2) (2000) 33–38
3. Maxion, R.A., Townsend, T.N.: Masquerade detection using truncated command lines. In: DSN '02: Proceedings of the 2002 International Conference on Dependable Systems and Networks, Washington, DC, USA, IEEE Computer Society (2002) 219–228
4. Posadas, R., Mex-Perera, J.C., Monroy, R., Nolazco-Flores, J.A.: Hybrid method for detecting masqueraders using session folding and hidden markov models. In Gelbukh, A.F., García, C.A.R., eds.: MICAI. Volume 4293 of Lecture Notes in Computer Science., Springer (2006) 622–631
5. Greenberg, S.: Using unix: Collected traces of 168 users. Research Report 88/333/45, Department of Computer Science, University of Calgary, Alberta, Canada (1988)
6. Maxion, R.A.: Masquerade detection using enriched command lines. dsn **00** (2003) 5
7. Lane, T., Brodley, C.E.: An application of machine learning to anomaly detection. In: In Proceedings of the 20th National Information Systems Security Conference. (1997) 366–380
8. Chinchani, R., Muthukrishnan, A., Chandrasekaran, M., Upadhyaya, S.: RACOON: Rapidly Generating User Command Data for Anomaly Detection from Customizable Templates. In: Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC '04), Tucson, Arizona (2004)

9. Welch, T.A.: A technique for high-performance data compression. IEEE Computer **17**(6) (1984) 8–19
10. Bertacchini, M., Fierens, P.I.: Preliminary results on masquerader detection using compression based similarity metrics. Electronic Journal of SADIO **7**(1) (2007)
11. Cilibrasi, R., Vitányi, P.: Clustering By Compression. IEEE Transactions on Information Theory **51**(4) (2005) 1523– 1545
12. Li, M., Vitányi, P.: An Introduction to Kolmogorov Complexity and Its Applications. Second edn. GTCS. pub-SV (1997)
13. Bertacchini, M., Benitez, C.: NCD based masquerader detection using enriched command lines. In: Proc. of the IV Congreso Iberoamericano de Seguridad Informatica (CIBSI'07), Mar del Plata, Argentina (2007) 329–338
14. Evans, S., Eiland, E., Markham, S., Impson, J., Laczo, A.: Mdlcompress for intrusion detection: Signature inference and masquerade attack. Military Communications Conference, 2007. MILCOM 2007. IEEE (2007) 1–7
15. Latendresse, M.: Masquerade detection via customized grammars. In Julisch, K., Krügel, C., eds.: DIMVA. Volume 3548 of Lecture Notes in Computer Science., Springer (2005) 141–159
16. Nevill-Manning, C.G., Witten, I.H.: Identifying hierarchical structure in sequences: a linear-time algorithm. Journal of Artificial Intelligence Research **7** (1997) 67–82
17. Oka, M., Oyama, Y., Kato, K.: Eigen co-occurrence matrix method for masquerade detection (2004)
18. Turk, M., Pentland, A.: Eigenfaces for recognition. J. Cognitive Neuroscience **3**(1) (1991) 71–86
19. Oka, M., Oyama, Y., Abe, H., Kato, K.: Anomaly detection using layered networks based on eigen co-occurrence matrix. In Jonsson, E., Valdes, A., Almgren, M., eds.: RAID. Volume 3224 of Lecture Notes in Computer Science., Springer (2004) 223–237
20. Chen, L., Dong, G.: Masquerader detection using oclep: One-class classification using length statistics of emerging patterns. In: Web-Age Information Management Workshops, 2006. WAIM '06. Seventh International Conference on. (2006) 5
21. Wang, K., Stolfo, S.J.: One-class training for masquerade detection (2003)
22. Böckler, S.B., Bateman, A.: An introduction to hidden markov models. Current protocols in bioinformatics / editoral board, Andreas D. Baxevanis ... [et al.] **Appendix 3** (2007)
23. Okamoto, T., Watanabe, T., Ishida, Y.: Towards an immunity-based system for detecting masqueraders. In: KES. (2003) 488–495
24. Okamoto, T., Watanabe, T., Ishida, Y.: Mechanism for generating immunity-based agents that detect masqueraders. In: KES. (2004) 534–540
25. Okamoto, T., Ishida, Y.: Framework of an immunity-based anomaly detection system for user behavior. In Apolloni, B., Howlett, R.J., Jain, L.C., eds.: KES (3). Volume 4694 of Lecture Notes in Computer Science., Springer (2007) 821–829
26. McCallum, A., Nigam, K.: A comparison of event models for naive bayes text classification (1998)
27. Maxion, R., Townsend, T.: Masquerade detection augmented with error analysis. Reliability, IEEE Transactions on **53**(1) (2004) 124–147
28. Killhourhy, K.S., Maxion, R.A.: Investigating a possible flaw in a masquerade detection system. Technical Report 869, Newcastle University, School of Computing Science (2004)
29. Jia, P., Maxion, R.A.: Detect masqueraders using unix command sequences. Lab Report, Center for Automated Learning and Discovery (2005)

30. Killourhy, K.S., Maxion, R.A.: Learning from a flaw in a naive-bayes masquerade detector (2007)
31. Yung, K.H.: Using feedback to improve masquerade detection. In: ACNS. (2003) 48–62
32. Yung, K.H.: Using self-consistent naive-bayes to detect masquerades. In: PAKDD. (2004) 329–340
33. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological) **39**(1) (1977) 1–38
34. Gebski, M., Wong, R.K.: Intrusion detection via analysis and modelling of user commands. In Tjoa, A.M., Trujillo, J., eds.: DaWaK. Volume 3589 of Lecture Notes in Computer Science., Springer (2005) 388–397
35. Coull, S., Branch, J., Szymanski, B., Breimer, E.: Intrusion detection: a bioinformatics approach. Computer Security Applications Conference, 2003. Proceedings. 19th Annual (2003) 24–33
36. Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. J. ACM **21**(1) (1974) 168–173
37. Coull, S.E., Szymanski, B.K.: Sequence alignment for masquerade detection. Comput. Stat. Data Anal. **52**(8) (2008) 4116–4131
38. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Min. Knowl. Discov. **2**(2) (1998) 121–167
39. Chen, L., Aritsugi, M.: An svm-based masquerade detection method with online update using co-occurrence matrix. In Büschkes, R., Laskov, P., eds.: DIMVA. Volume 4064 of Lecture Notes in Computer Science., Springer (2006) 37–53
40. Li, Z., Li, Z., Liu, B.: Masquerade detection system based on correlation eigen matrix and support vector machine. Computational Intelligence and Security, 2006 International Conference on **1** (2006) 625–628
41. Kim, H.S., Cha, S.D.: Empirical evaluation of svm-based masquerade detection using unix commands. Computers and Security **24**(2) (2005) 160–168
42. Yang, M., Zhang, H., Cai, H.: Masquerade detection using string kernels. Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on (2007) 3681–3684
43. Wan, M.D., Wu, H.C., Kuo, Y.W., Marshall, J., Huang, S.H.: Detecting masqueraders using high frequency commands as signatures. Advanced Information Networking and Applications - Workshops, 2008. AINAW 2008. 22nd International Conference on (2008) 596–601
44. Wang, W., Guan, X., Zhang, X.: Profiling program and user behaviors for anomaly intrusion detection based on non-negative matrix factorization. 43rd IEEE Conference on Decision and Control (2004)
45. Mex-perera, C., Posadas, R., Nolazco, J.A., Monroy, R., Soberanes, A., Trejo, L.: An improved non-negative matrix factorization method for masquerade detection. In: Proceedings of the 1st Mexican International Conference on Informatics Security, MCIS 2006, Mexico (2006)
46. Beauquier, J., Hu, Y.J.: Intrusion detection based on distance combination. In: CESSE07, Venice, Italy, World Acacemy of Sciences, WAS (2007)