Masquerade Mimicry Attack Detection: A Randomised Approach☆

Juan E. Tapiador*, John A. Clark

Department of Computer Science, University of York, Deramore Lane, York, YO10 5GH, UK

Abstract

A masquerader is an (often external) attacker who, after succeeding in obtaining a legitimate user's credentials, attempts to use the stolen identity to carry out malicious actions. Automatic detection of masquerading attacks is generally undertaken by approaching the problem from an anomaly detection perspective: a model of normal behaviour for each user is constructed and significant departures from it are identified as potential masquerading attempts. One potential vulnerability of these schemes lies in the fact that anomaly detection algorithms are generally susceptible to deception. In this work, we first investigate how a resourceful masquerader can successfully evade detection while still accomplishing his goals. For this, we introduce the concept of masquerade mimicry attacks, consisting of carefully constructed attacks that are not identified as anomalous. We then explore two different detection schemes to thwart such attacks. We first study the introduction of a blind randomisation strategy into a baseline anomaly detector. We then propose a more accurate algorithm, called Probabilistic Padding Identification (PPI) and based on the Kullback-Leibler divergence, which attempts to identify if a sufficiently anomalous attack is present within an apparently normal behavioural pattern. Our experimental results indicate that the PPI algorithm achieves considerably better detection quality than both blind randomised strategies and adversarial-unaware approaches.

Keywords: anomaly detection, insider threats, masqueraders, mimicry attacks, Kullback-Leibler divergence

☆A preliminary version of this paper appeared in the Proceedings of the 4th IEEE Conference on Network and System Security (NSS 2010) [43].
This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

*Corresponding author.
Email addresses: `jet@cs.york.ac.uk` (Juan E. Tapiador), `jac@cs.york.ac.uk` (John A. Clark)

1. Introduction

One of the worst threats in computer security is that posed by internal users who misuse their privileges for malicious purposes. Such actions could potentially result in enormous damages for an organisation, arguably far greater than those expected from external adversaries. Classical access control models can partially alleviate the risks associated with internal security issues, but the reality of many systems is unfortunately quite complex [22]: specifying good security policies is very hard; policies are frequently and purposely bypassed to get the job done; sharing information among different organisations is too often necessary and current security models are very poor at controlling the potential repercussions of wrong-sharing; etc. As a consequence, it has been recognised that access control systems are necessary measures, but clearly insufficient to deal with all the complexities posed by insider attacks. Research in this area has been in place for the last 20 years and, to some extent, has proliferated lately; see e.g. [35, 4, 3, 8] for a few examples of recently reported research initiatives.

One traditional way of classifying insiders is as *traitors* and *masqueraders* [37]. A traitor is a user who already enjoys some privileges within the system and whose purposes will affect negatively the security properties of the organisation's information and systems. A masquerader, on the contrary, is an often external attacker who succeeds in obtaining a legitimate user's credentials and attempts to use the stolen identity to carry out malicious actions (e.g. credit card fraudsters).

Virtually all existing masquerade detection approaches rely upon one key observation: *"behaviour is not something that can be easily stolen"* [37]. Profiling users behaviours could therefore establish models of normalcy such that deviations from them would presumably indicate the presence of an impersonation attempt. The idea of using anomalies as proxies for attacks has been extensively studied in various security domains and, albeit generally useful, is not free from drawbacks and controversies [40]. Furthermore, there are inherent limitations in using an anomaly detection algorithm as the basis for masquerade detection. Firstly, profiles are ultimately derived from data provided by the user, who might well be in the business of forcing the learning process to build something undesirable, such as for example a model of normalcy such that future misbehaviours will not be identified. Some works [24, 7] have already pointed out that the data used to train a security application could be actively manipulated by an adversary. When applied to such adversarial domains, learning algorithms should be conveniently adapted, but research in this area is still scarce. A second threat stems from the fact that knowledge of some details about the detection process facilitates evasion. Yet in general it is reasonable to assume that such information is public, as it is in general possible for an adversary to obtain it by careful experimentation with the system [29].

1.1. Our Contributions

In this paper we investigate some of the threats posed by sophisticated attackers in the context of masquerade detection. In particular, we introduce the concept of masquerade mimicry attacks:

Definition 1. *A* ***masquerade mimicry attack*** *is an attack where an impersonator attempts to evade being detected by a deployed masquerade sensor. Such attacks work by modifying the original attack pattern exhibited by the impersonator in such a way that the resulting behaviour looks normal, i.e., as belonging to the user being impersonated.*

We make the following specific contributions:

1. We demonstrate masquerade mimicry attacks against One-Class Naïve Bayes (OCNB), a widely used masquerade detection algorithm. In particular, we provide concrete procedures for generating such attacks and evaluate empirically their effectiveness using a real-world dataset. Moreover, the algorithm given here for generating mimicry attacks is valid not only for OCNB, but also for a larger class of detectors.
2. We describe and evaluate a randomised variant of OCNB based on the use of multiple random bags (OCNB-MRB). The use of randomised classifiers has proven useful in other applications. In this case, our results suggest that OCNB-MRB achieves a considerable improvement in detection accuracy, but many attacks still go unnoticed.
3. In order to improve upon OCNB-MRB, we propose and evaluate a novel detection mechanism based on the idea of separating, in a probabilistic sense, the attack from the padding sequence in a block of data. The proposed algorithm, called Probabilistic Padding Identification (PPI), makes use of the Kullback-Leibler divergence and does not rely on any assumptions about the attack other than, once isolated, it is anomalous. We empirically demonstrate the improvement achieved through this method in terms of detection quality.

1.2. Organisation

The rest of this paper is organised as follows. In Section 2 we discuss previous work on masquerade detection and mimicry attacks. In Section 3 we describe the OCNB masquerade detection algorithm, which will be used throughout this paper to illustrate our contributions. Section 4 introduces mimicry attacks in the context of a masquerade detection scenario. We describe various methods for generating such attacks and empirically evaluate their success in evading detection. In Section 5 we explore the use of a randomised version of OCNB to counteract such attacks. In Section 6 we describe and evaluate an alternative method called the PPI algorithm. The results obtained over a dataset containing normal samples, as well as mimicry and non-mimicry masquerade attacks, are shown in Section 7. Finally, Section 8 concludes the paper by highlighting our main contributions and discussing some avenues for future research.

2. Related Work

In this section we review the two research areas most related to our work, namely masquerade detection algorithms and the concept of mimicry attacks in other contexts.

2.1. Masquerade Detection

Schonlau *et al.* presented in [39] the problem of differentiating between users conducting their normal activity and those who have been impersonated by an attacker. The work introduced a dataset[1] for the evaluation of different masquerade detection methods. The dataset consists of sequences of truncated UNIX commands corresponding to the normal activity of 70 users and collected over a period of several months. Users' activities are grouped into blocks of 100 consecutive commands, and the main task for a masquerade detection algorithm is to accurately identify non-self blocks as anomalous (and, therefore, implicitly mark them as masquerade attempts), while correctly classifying the self blocks as belonging to the user. The work in [39] explores the performance of six different machine learning algorithms for this task in the so-called SEA configuration: each user's first 5000 commands are used for training and the remaining 10000 commands for testing on a per-block basis.

A series of papers by Maxion *et al.* improved on the results reported in [39] and provided further analysis of the masquerade detection problem. In [32] it is shown how the naïve Bayes classifier achieves much better performance than previously proposed schemes. The paper also provides an excellent articulation of why some users are more difficult to attack than others and introduces a new experimental setting called 1v49, as opposed to the original SEA experiment described in [39]. The 1v49 experiment is arguably a better way of evaluating the performance of detection algorithms. We refer the reader to [32] for additional information.

Further work explored the consequences of using datasets enriched with information other than commands alone [33], as well as the effects of applying privacy-preserving sanitisation strategies over the data [25]. Wang and Stolfo argued in [46] that detection methods based on one-class training (i.e., relying only on self data) are more appropriate for a real-world setting. They showed that naïve Bayes and Support Vector Machine (SVM) algorithms attain similar results both in a one-class configuration and by using two-class data.

Work on masquerade detection, and more generally on profiling user behaviour for security purposes, has proliferated over the last decade, especially concerning the study of different detection strategies. Some of the proposals include information-theoretic approaches [1, 12], hidden Markov models [36], or sequence- and text-mining [34, 28, 6, 18] schemes, among others. Despite the diversity of principles behind these methods, the reported results show that they all perform similarly in terms of accuracy.

[1] Publicly available at `http://www.schonlau.net`.

2.2. Mimicry Attacks

The notion of *mimicry* is generally taken from Biology [9] and indicates the process of intentionally altering the appearance or behaviour of an entity with the purpose of inducing an error in an observer. In computer and network security, the basic idea behind mimicry attacks is to evade an anomaly detector by altering the attack to make it look normal. Evasion is successful when the modified data block being analysed fit the normal profile used by the detector, while simultaneously preserving the intended goal of the attack. Introducing such transformations generally requires the attacker to know both the detection algorithm and the model of normalcy in use.

Early work on mimicry attacks targeted host-based IDSs, in particular systems based on the analysis of system call sequences as introduced by Forrest *et al.* [15, 16, 21, 49]. Wagner *et al.* [44, 45] and Tan *et al.* [41, 42] developed various strategies for generating mimicry attacks against such detectors. Subsequent work, such as e.g. [17, 19, 26, 23], further explored this idea, mainly focusing on the problem of how to generate a mimicry sequence that evades detection and achieves the attacker's goals. The task is generally computationally hard, and techniques drawn from domains such as model checking, code analysis, or genetic programming have proven useful.

Similar ideas have also been investigated in the area of network-based IDS, where detection is accomplished by analysing payload features such as byte distributions or, more generally, n-gram or more complex models such as in [47, 48, 27, 30, 31, 10, 11]. Fogla *et al.* introduced in [13, 14] polymorphic blending attacks, where the main idea is to generate each attack instance in such a way that its statistics match the profile of normalcy used by an anomaly detector. Such attacks would therefore be able to evade both signature- and anomaly-based IDSs. Again, it is shown that the problem of generating such instances is NP-complete, though some heuristic techniques are of help.

To the best of our knowledge, no previous work has explored the existence of mimicry attacks in the context of masquerade detection, as well as suitable countermeasures. These are the main goals of this paper.

3. One-Class Naïve Bayes (OCNB) Masquerade Detection

In this section we describe a widely-used masquerade detection algorithm, the One-Class Naïve Bayes (OCNB), which will be extensively used later to demonstrate masquerade mimicry attacks.

The naïve Bayes (NB) classifier [20] is a supervised learning algorithm which has been used in a wide range of applications. NB is often a very attractive solution because of its simplicity, efficiency and excellent performance. It uses the Bayes rule to estimate the probability that an instance $x = (x_1, \ldots, x_m)$ belongs to class y as

$$P(y|x) = \frac{P(y)}{P(x)} P(x|y) = \frac{P(y)}{P(x)} \prod_{i=1}^{m} P(x_i|y) \tag{1}$$

so the class with highest $P(y|x)$ is predicted. (Note that $P(x)$ is independent of the class and therefore can be omitted.) The naïvety comes from the assumption that in the underlying probabilistic model all the features are independent, and hence $P(x|y) = \prod_{i=1}^{m} P(x_i|y)$.

NB has been used in the context of masquerade detection [32, 46], particularly using Schonlau *et al.*'s dataset. In the multinomial model (or bag-of-words approach), every block of commands B to be classified is represented by a vector of attributes $[n_1(B), \ldots, n_m(B)]$, where $n_i(B)$ is the number of times command c_i appears in the block. The probability $P(y|B)$ given by (1) can be then computed as

$$P(y|B) = P(y) \prod_{i=1}^{m} P(c_i|y)^{n_i(B)} \tag{2}$$

The probabilities $P(c_i|y)$ are derived from a training set consisting of labelled instances for all possible classes (e.g., from each user's first 5000 commands in Schonlau *et al.*'s dataset), and the priors $P(y)$ are often ignored. In order to control the sensitivity to previously unseen commands, it is convenient to ensure that all commands appear with non-zero probability even if some of them are not present at all in the training set. This can be achieved by using an additive smoothing over the estimated probabilities

$$P(c_i|y) = \frac{\sum_{B \in \mathcal{T}(y)} n_i(B) + \alpha}{|B| \cdot |\mathcal{T}(y)| + \alpha \cdot m} \tag{3}$$

where $\mathcal{T}(y)$ is the training set for class y and α the smoothing parameter.

For convenience, in this work we will use minus the logarithm of (2) rather than the raw probability as basic indicator of the nature of a block (again, ignoring the priors):

$$score(B) = -\log P(y|B) = -\sum_{i=1}^{m} n_i(B) \log P(c_i|y) \tag{4}$$

The result can be seen as an anomaly score: the higher its value, the more anomalous the block is, and vice versa.

Following [46], in a one-class (OC) setting the training set for each user consists exclusively of data corresponding to self activities. Since a profile of non-self behaviour is not required, the detection is performed by simply comparing the probability of a block being self (or, equivalently, the anomaly score) to a threshold. Such a threshold can be adjusted to control the false and true positive rates, and the resulting ROC (Receiver Operating Characteristic) curve provides a way of measuring the detection quality. Different ROC curves can be compared by computing the Area Under the Curve (AUC), also known as the ROC score: An AUC close to 1 indicates near optimal detection quality, and vice versa. Figure 1 shows the AUC for each one of the 50 users in the Schonlau *et al.*'s dataset using OCNB in the 1v49 experimental setting. These results (or similar ones obtained with different detection methods) have been previously

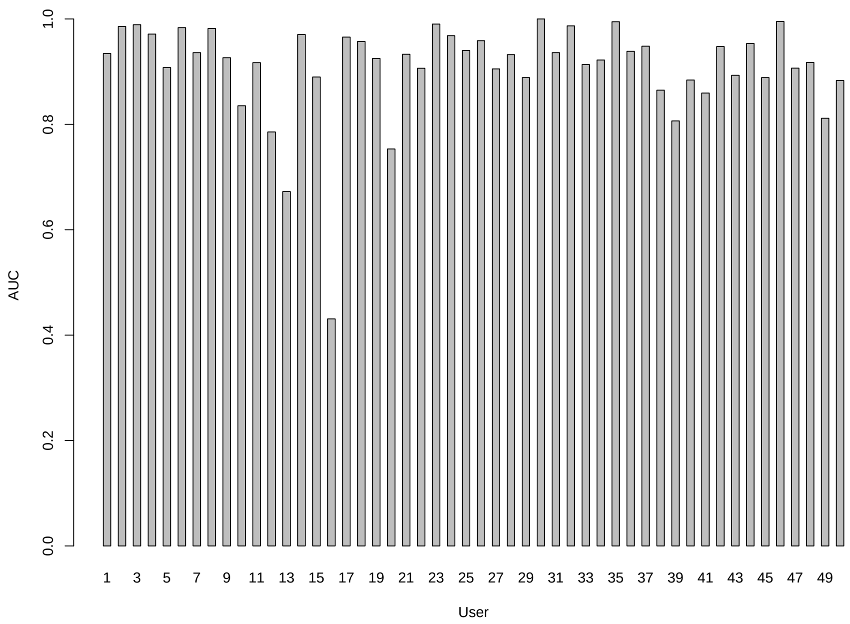

Figure 1: AUCs for the 50 users in the Schonlau *et al.*'s dataset using OCNB and the 1v49 experiment.

reported, e.g. in [46, 38], and we reproduce them here for completeness. It can be observed how OCNB achieves fairly good detection results in most cases, although some users (e.g. 13 and 16) are more easy to impersonate than others. A detailed analysis can be found in [32].

4. Masquerade Mimicry Attacks

In this section we introduce mimicry attacks in the context of a masquerade detection problem. We consider an adversary who intends to launch an attack consisting of a sequence of actions or commands. We make three fundamental assumptions about this process:

(i) *Perfect knowledge:* The adversary knows perfectly the detection algorithm being used and all the relevant parameters, as well as the model of normalcy for the user whose system account is impersonating. Alternatively, the adversary could be the user himself attempting to launch an attack without being spotted by the anomaly detector.

(ii) *Non-poisoned detector:* The detector has been trained with attack-free data, so we do not consider the possibility of frog-boiling attacks (e.g. [5])

or other forms of evasion based on training the detection algorithm with carefully crafted data.

(iii) *Attack padding:* The attack sequence must be executed within a block, but not necessarily in a contiguous way. Thus, the adversary could insert padding commands at any point of the attack sequence. We do not put any restriction on the type, length, position, or number of padding sequences, other than both attack and padding must add up to a block size.

4.1. Notation

We will denote sequences or blocks of commands by capital letters, in particular A for attacks, P for padding, and B for entire blocks. The symbol $|\cdot|$ denotes the length of a sequence. Sequences will be treated as arrays, so $S(i)$ denotes the i-th command in the sequence. The probability density function of a sequence will be specified by a calligraphic font, e.g., $\mathcal{A}$, $\mathcal{P}$, $\mathcal{B}$, etc. Thus, $\mathcal{S}(c_i)$ will denote the frequency of command c_i in sequence S.

4.2. Evading OCNB

Consider an attack consisting of $|A| \leq |B|$ commands, so the number of padding commands the adversary must generate is $|B|-|A|$. We assume that the attack sequence will contribute significantly to identify the block as anomalous. For example, in the case of a detector based on the OCNB classifier described above, this translates into a very low probability induced by the commands comprising the attack. In this case, the optimal padding strategy for the attacker consists of filling the block with the command $c_{max} = \arg\max_{c_i} \mathcal{M}(c_i)$, $\mathcal{M}$ being the model of normalcy, as this will cause the maximum possible increment in the probability of the block being classified as normal given the attack. Despite being optimal against OCNB, we will not consider such a strategy here since the results might not be generally useful for different detection algorithms. We shall instead look into the more general strategy of producing a padding sequence such that the histogram of the resulting block (attack plus padding) is statistically indistinguishable from that observed during training. Such attacks would be presumably effective against a wider range of masquerade detection algorithms.

4.2.1. Attack Generation

We will assume that the distinguishability metric we attempt to minimise is $\sum_{c_i} |\mathcal{B}(c_i) - \mathcal{M}(c_i)|$, where $\mathcal{B}$ and $\mathcal{M}$ are the histogram of the block and the normalcy model, respectively, and the sum is taken over the available set of commands. We will also restrict ourselves to the case where the attack sequence is immutable, i.e. no command in it can be deleted or replaced by other. In this case, it is not difficult to see that the optimal strategy for generating the padding sequence consists of:

(i) Compute the difference histogram: $\mathcal{D}(c_i) = \mathcal{M}(c_i) - \mathcal{A}(c_i)$ if $\mathcal{M}(c_i) \geq \mathcal{A}(c_i)$, and $\mathcal{D}(c_i) = 0$ otherwise.

(ii) Add to the padding sequence $|B| \cdot \mathcal{D}(c_m)$ instances of the command $c_m = \arg\max_{c_i} \mathcal{D}(c_i)$.

lpdsend grep date $\boxed{\texttt{cpp}}$ lp find expr generic mp $\boxed{\texttt{sh}}$ file post $\boxed{\texttt{xrdb}}$ awk
rm ln getpgrp $\boxed{\texttt{mkpts}}$ LOCK ls $\boxed{\texttt{env}}$ sed FIFO gethost $\boxed{\texttt{csh}}$ download $\boxed{\texttt{kill}}$
$\boxed{\texttt{userenv}}$ tcpostio UNLOCK rmdir tcppost $\boxed{\texttt{wait4wm}}$ mimencod MediaMai netstat
$\boxed{\texttt{xhost}}$ netscape popper gettxt $\boxed{\texttt{xsetroot}}$ xconfirm endsessi tellwm $\boxed{\texttt{reaper}}$
xprop xdm $\boxed{\texttt{cat}}$ toolches 4Dwm xterm xwsh sendmail $\boxed{\texttt{mail}}$ gs xdvi.rea xdvi
last dc imgview $\boxed{\texttt{launchef}}$ xv .wrapper uname fmarch .maker_w maker5X.
$\boxed{\texttt{hostname}}$ .java_wr dirname basename egrep java make acroread $\boxed{\texttt{ps}}$ cal xcal
touch nslookup unpack $\boxed{\texttt{id}}$ col ul more man ping finger emacs-20 $\boxed{\texttt{nawk}}$
PLATFORM Slmhelpe ftp wc mkdir $\boxed{\texttt{getopt}}$ lpdsend tektroni dev.moti Sqpe

Figure 2: Example of masquerade mimicry attack. Framed commands correspond to an attack sequence of length 20; the remaining 80 commands (padding) are generated to fit User 0's profile.

(iii) Set $\mathcal{D}(c_m) = 0$ and repeat step (ii) until no more padding is needed.

Alternatively, a suboptimal (but certainly much faster) strategy consists of generating the padding by just sampling from the difference distribution $\mathcal{D}$. (The procedure is straightforward once the inverse of cumulative distribution, $F_{\mathcal{D}}^{-1}$, is computed.)

To build the final block of commands, we first select $|A|$ different random positions of the block and place one attack command in each of them, respecting the original order in the attack sequence. The remaining empty positions are then filled up with the padding commands previously generated in no particular order. Figure 2 shows an example.

4.2.2. Results

In order to quantify the performance of such attacks, we have conducted the following experiment using the Schonlau *et al.*'s dataset. Given a user u, we first repeat the 1v49 experiment and record the raw scores issued by OCNB. We then plot the distribution of the scores for both self and non-self blocks. This serves to visually illustrate the discriminative capability of the classifier: the higher the overlapping between both distributions, the lower the detection quality. As an example, Fig. 3 shows the distribution of the scores given by OCNB to user 2's self and non-self blocks (two leftmost boxplots).

"Attacks" are generated by randomly choosing a sequence of $|A|$ commands from a block belonging to the training dataset of a user other than u. Note that such sequences are not by any means *actual* attacks. However, our emphasis here is not on the consequences of the adversary's actions in a real setting, but rather on the assumption that attacks are anomalous events which nonetheless might be conveniently camouflaged to avoid detection. For this purpose, the methodology here followed should do as far as the detection of such *concealed anomalies* is concerned. This sequence is then placed into an empty block, and the remaining $100 - |A|$ positions are filled with a padding sequence obtained by following the

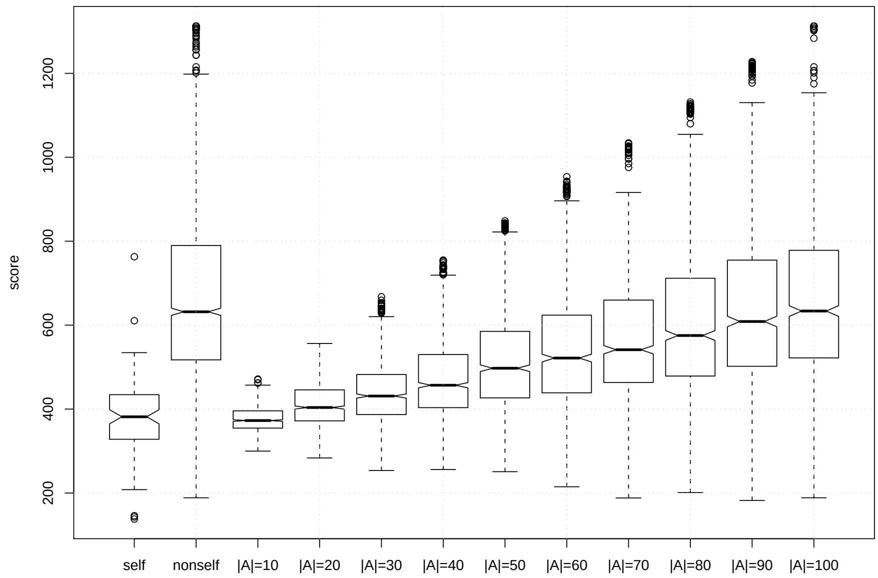

Figure 3: Distribution of OCNB scores for user 1 including mimicry attacks of various lengths.

optimal strategy described above. The score for the block as given by OCNB is computed and the procedure is repeated 10000 times for randomly generated attacks. The ten rightmost boxplots in Fig. 3 show the score distribution for attacks of length 10, 20, . . ., 100. It is observed that the bulks of the self and non-self distributions are largely non-overlapping, and a threshold around 500 might serve to detect most nonself sequences with some rate of false positives and negatives. Mimicry attacks (ten rightmost plots) of low length present a score distribution below any reasonable detection threshold, thus being essentially impossible to detect. An increasing attack length generates more anomalies per block and also leaves less space available for padding, which translates into a greater score and, consequently, more chances of detection. The plots for most users are completely analogous.

In global terms, OCNB performs rather poorly in detecting this form of attacks. Table 1 gives the average detection rate of mimicry attacks of length up to 60 commands computed for the 50 users in the dataset. The detector for each user was tuned so as to limit the false positive rate to a maximum of 5%, and the average is computed for the 50 users. The majority of the attack blocks passed unnoticed by the detector, only approaching a detection rate higher than 50% (which is still remarkably low) when the attack sequence comprises more than half the block length.

Table 1: Average detection rate of mimicry attacks using OCNB.

Attack length	$\|A\| = 10$	$\|A\| = 20$	$\|A\| = 30$	$\|A\| = 40$	$\|A\| = 50$	$\|A\| = 60$
Avg. DR	0.081	0.206	0.314	0.407	0.474	0.521

4.3. Discussion

The results discussed above show the effectiveness of mimicry attacks to evade OCNB and, presumably, many others masquerade detectors. In a way, this does not come as a surprise, as none of these algorithms were designed to operate in the adverse conditions imposed by sophisticated attackers. This fact alone motivates the need for adversarial-aware classifiers, that is, algorithms factoring in the possibility of an intelligent adversary manipulating the input. In the remaining of this paper we introduce and study two alternative methods to tackle this question.

5. OCNB with Multiple Random Bags

One simple way of reducing the attacker's chances of successfully evading a classifier is through randomisation [2, 7]. By introducing a probabilistic component into the detection process, the attacker will inevitably lose some degree of control over the effect of his actions on the classification outcome. Unfortunately, this will also influence negatively the overall detection performance, particularly in terms of a potentially higher rate of false positives, and therefore should be done carefully.

OCNB admits an easy and elegant randomisation strategy by using the so-called *Multiple Random Bags* (MRB) approach. Recall that OCNB works by computing an anomaly score (essentially a probability) given a block $B = \{c_1, \ldots, c_n\}$. The idea here consists of splitting B into k randomly selected smaller blocks, called bags, B_i, each one of size $\ell < |B|$. The overall anomaly score of the block is then computed as

$$score(B) = \max\{score(B_i)\}_{i=1}^{k} \tag{5}$$

The intuition behind this scheme is simple. If a block is entirely normal, so it will be any randomly selected subset given appropriate parameters. Conversely, if a block contains an attack camouflaged among normal commands, perhaps one of the randomly chosen samples may contain a significant amount of attack commands. As the overall anomaly score is that of the most anomalous bag, the chances of correctly identifying a mimicry attack increase with the number of bags k. As for the optimal bag length ℓ, it is obviously related to the attack length we attempt to spot, with low values generally leading to better detection rates. There is however a trade-off here, since too small bags may break down users' behavioural patterns and increase the false positive rate. The interested reader can find in [50] a similar idea applied to the spam detection setting.

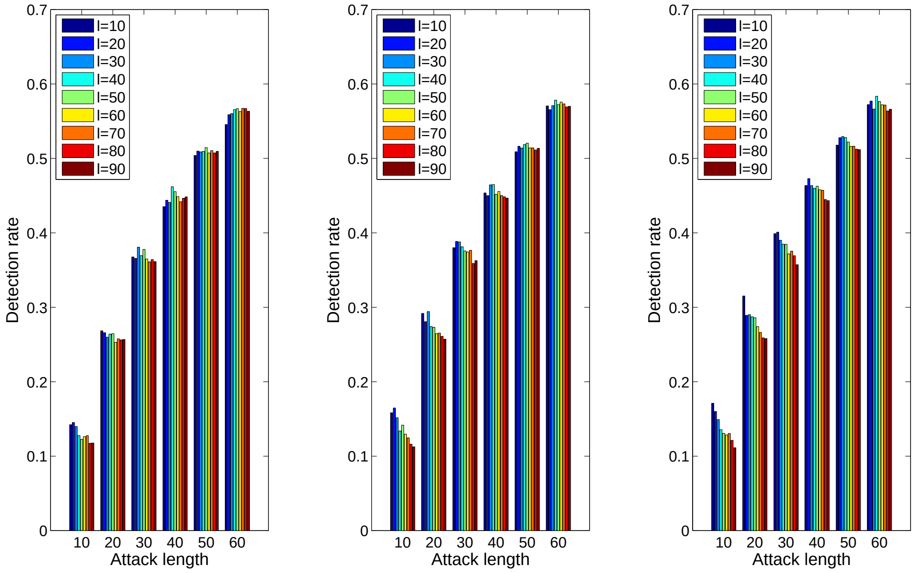

Figure 4: Detection rate of masquerade mimicry attack using OCNB-MRB with $k = 5$ (left), $k = 10$ (centre) and $k = 25$ (right).

5.1. Experimental results

We have repeated the experiments described in Section 4.2 but using OCNB with MRB. On a first set of experiments, we investigate the effect of parameters k and ℓ on the detection performance against masquerade mimicry attacks. Figure 4 shows the detection rate achieved for $k = 5$, 10 and 25. For each value, we study values of $\ell = 10, 20, \ldots, 90$ and different attach lengths. As it can be observed, the use of MRB improves upon the detection rates obtained by OCNB (compare with the values reported in Table 1), although not spectacularly. On average, the MRB approach achieves around 8-10% more in terms of successful detection, with generally better values for attacks of short length.

In terms of parameterisation, the trend observed in our experiments is quite clear: The more the number of bags (k), the better the detection rate. There is a simple explanation for this: Each random bag can be seen as an independent experiment where a number of samples are taken from the block, and its anomaly score is then computed. The more the number of experiments, the higher the chances of getting a bag with a number of attack commands sufficient to spot the block as anomalous. A bigger number of bags will, of course, increase the time required to carry out the detection. We will address this issue later. As for the bag length ℓ, the behaviour seems to be different depending on the attack length. Smaller bags perform better for short attacks. This, again, is reasonable and

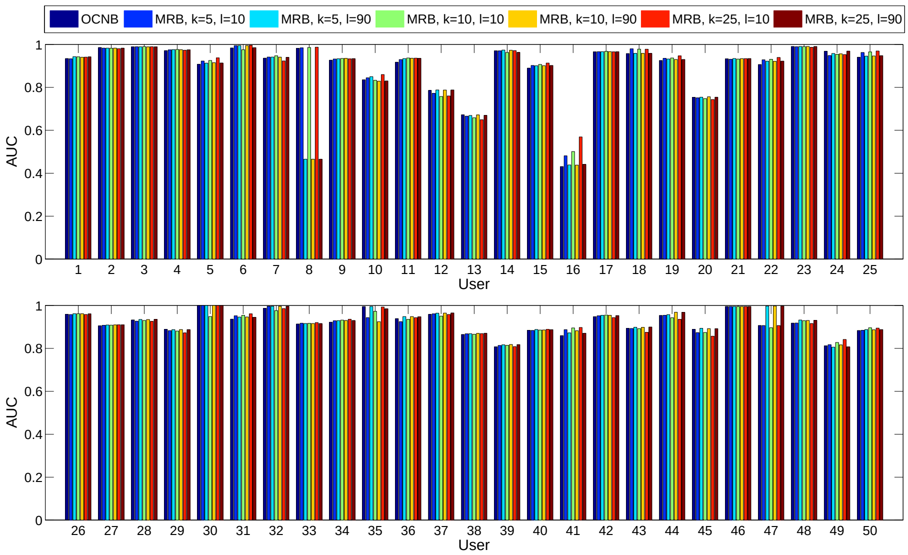

Figure 5: AUCs for the 50 users in the Schonlau *et al.*'s dataset using OCNB-MRB and the 1v49 experiment. For comparison, the AUCs obtained with OCNB are also provided.

conform with our intuition: if the attack sequence is very low compared with the bag size, each random bag will contain far more normal commands than attack ones, and therefore the anomaly score will tend to be low. In the case of long attacks (say, $|A| = 60$ and higher), this relation is not obvious and bags of almost any length suffice to detect most attacks.

It remains to be seen whether or not using MRB has a negative effect in terms of false negatives, and also how it performs against usual, non-mimicry masquerade attacks. In order to evaluate this we have repeated the 1v49 experiment but using OCNB-MRB. Figure 5 shows the original AUCs obtained with OCNB and the ones corresponding to MRB with different values of parameters k and ℓ. In most cases, the use of MRB has no adverse impact whatsoever in the ROC curves, and the AUCs are almost identical to those obtained with OCNB. In fact, for a few users employing MRB helps to reduce slightly the number of false positives: see e.g. users 11, 16, and 47.

The use of MRB does not impose any noticeable burden to the overall detection process. Table 2 shows the average time required to process a 100 command block and compute its anomaly score. These experiments were carried out in a laptop with an Intel Core i7 at 2.66 GHz (2 cores) and 8 GB of memory. It can be seen how both OCNB and the MRB variant are reasonably fast. In the case of MRB, the processing times increases approximately linearly both with k and ℓ. In any case, within the range of parameters values here explored, the total time never exceeds a fraction of a millisecond.

Table 2: OCNB and OCNB-MRB processing times per 100 command block.

Algorithm	Time in ms (Avg. $\pm$ Std. Dev.)
OCNB	0.0024 ± 0.0005
OCNB-MRB ($k = 5$, $\ell = 10$)	0.0056 ± 0.0013
OCNB-MRB ($k = 5$, $\ell = 90$)	0.0495 ± 0.0033
OCNB-MRB ($k = 10$, $\ell = 10$)	0.0108 ± 0.0017
OCNB-MRB ($k = 10$, $\ell = 90$)	0.0996 ± 0.0071
OCNB-MRB ($k = 25$, $\ell = 10$)	0.0263 ± 0.0022
OCNB-MRB ($k = 25$, $\ell = 90$)	0.2438 ± 0.0073

6. Probabilistic Padding Identification (PPI)

In this section we try to improve on the results obtained with OCNB-MRB by using a more elaborate strategy. We next develop an algorithm which attempts to separate the attack from the padding sequence in a given block of commands. The process will be carried out with the help of the normalcy model presumably used to generate the padding, but without any further knowledge about the attack length (which, incidentally, could be zero). We first review some properties of the Kullback-Leibler divergence, a concept which will be central in our algorithm.

6.1. Kullback-Leibler Divergence

The Kullback-Leibler (KL) divergence is a non-symmetric measure of the difference between two probability distributions. If P and Q are two discrete distributions, then the KL divergence of Q from P is defined by

$$D_{KL}(P \parallel Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \tag{6}$$

Note that $D_{KL}(P \parallel Q)$ can be rewritten as

$$\begin{aligned} D_{KL}(P \parallel Q) &= -\sum_i P(i) \log Q(i) + \sum_i P(i) \log P(i) \\ &= H(P, Q) - H(P) \end{aligned} \tag{7}$$

where H denotes the entropy. Consequently, D_{KL} admits a simple interpretation as the expected number of extra bits necessary to encode samples taken from P when using a code based on Q rather than one based on P.

From a different perspective, the KL divergence can also be seen as the expected discrimination information between two hypothesis. Given a sample x and two possible hypothesis H_0 and H_1, $D_{KL}(P(x|H_1) \parallel P(x|H_0))$ provides the mean information per sample for discriminating in favour of H_1 against H_0, given that H_1 is true. Or, in other words, it measures as the amount of evidence for H_1 over H_0 to be expected per sample.

6.2. The PPI Algorithm

Based on the properties of the KL divergence, we next describe an algorithm to probabilistically identify the padding portion of a block of commands. Assume that A and P are the attack and padding portions of a block B, and assume that $\mathcal{M}$ is the normalcy model for a given user. The algorithm relies upon two main observations:

(i) $\mathcal{A}$ is sufficiently different from $\mathcal{M}$ (otherwise it would not be necessary to add padding); and

(ii) $\mathcal{P}$ is highly similar to $\mathcal{M}$, as it has to compensate for the effects of $\mathcal{A}$.

Note that the problem of extracting P from B is further complicated by the fact that we generally do not know the length of the attack.

Our approach consists of identifying subsets $\hat{P}$, $\hat{A} \subseteq B$, with $\hat{P} \cup \hat{A} = B$ and $\hat{P} \cap \hat{A} = \emptyset$, such that $D_{KL}(\hat{\mathcal{P}} \parallel \mathcal{M})$ is very low and, simultaneously, $D_{KL}(\hat{\mathcal{A}} \parallel \mathcal{M})$ is very high. An exhaustive search would require to check $2^{|B|}$ possible subsets and compute two KL divergences for each one of them, which is clearly impractical. Instead, we propose a greedy strategy where suitable candidates for $\hat{P}$ and $\hat{A}$ are identified in one single pass over the block.

The algorithm, shown in Fig. 6, attempts to identify the portion $\hat{P}$ of B that best fits the model. A vector C is used to indicate whether command $B(i)$ is padding or not, so at each step such a vector partitions the block into two sequences, $\hat{P}$ and $\hat{A}$. The procedure DiffKL computes the KL divergences between each of these sequences and the model $\mathcal{M}$, and returns the absolute value of the difference. At each step, the PPI algorithm is governed by a simple rule: add the i-th command to the tentative padding if, by doing so, the increment of the differential KL divergence is greater than that obtained by not adding the command. The rationale behind such a rule can be better understood by observing that

$$\begin{aligned} |D_p - D_a| &= \left| \sum_i \hat{\mathcal{P}} \log \frac{\hat{\mathcal{P}}}{\mathcal{M}} - \sum_i \hat{\mathcal{A}} \log \frac{\hat{\mathcal{A}}}{\mathcal{M}} \right| \\ &= \left| H(\hat{\mathcal{A}}) - H(\hat{\mathcal{P}}) + H(\hat{\mathcal{P}} - \hat{\mathcal{A}}, \mathcal{M}) \right| \end{aligned} \tag{8}$$

i.e., a command is accepted as belonging to padding if that translates into a higher difference of the entropies of $\hat{\mathcal{P}}$ and $\hat{\mathcal{A}}$, plus a higher difference in the cross entropy between $(\hat{\mathcal{A}} - \hat{\mathcal{P}})$ and the model $\mathcal{M}$. Implicit in this utility function is the idea that padding and attack have different *information content*, hence its use to identify both of them.

A simpler and more natural approach would appear to be to accept the i-th command as padding if that decreases the KL divergence between the candidate $\hat{\mathcal{P}}$ and $\mathcal{M}$. This alternative, to which we will refer as PPI KL as opposed to the previously discussed PPI DiffKL, turns out to be less effective in practice. We next discuss some experimental results.

Algorithm 1 PPI

Input: Block B, model $\mathcal{M}$
Output: Boolean vector: $C(i) =$ **true** if $B(i)$ is padding

Initially $C(i) \leftarrow$ **false** for all i
for $i = 1$ **to** $|B|$ **do**
    $\bar{d} =$ DiffKL($C, B, \mathcal{M}$)
    $C(i) \leftarrow$ **true**
    $d =$ DiffKL($C, B, \mathcal{M}$)
    if $d \leq \bar{d}$ **then**
        $C(i) \leftarrow$ **false**
    end if
end for
return $P =$ commands $B(i)$ such that $C(i)$ is **true**

Algorithm 2 DiffKL

Input: Boolean vector C, block B, model $\mathcal{M}$
Output: Difference of K-L divergences

$\hat{\mathcal{A}} \leftarrow$ PDF of those $B(i)$ such that $C(i)$ is **false**
$\hat{\mathcal{P}} \leftarrow$ PDF of those $B(i)$ such that $C(i)$ is **true**
$D_a \leftarrow D_{KL}(\hat{\mathcal{A}} \parallel \mathcal{M})$
$D_p \leftarrow D_{KL}(\hat{\mathcal{P}} \parallel \mathcal{M})$
return $|D_p - D_a|$

Figure 6: Probabilistic Padding Identification (PPI) algorithm.

6.3. Experimental Results

We now report results of the evaluation of the PPI algorithm over masquerade mimicry attacks only. Next section provides details on the overall behaviour over a dataset composed of both attacks and self samples.

For each possible attack length from 1 to 100, we have generated 10000 mimicry attacks following the procedure described in Section 4.2. Each attack is analysed by the PPI algorithm, which returns the estimated positions of the padding. We then compute how many true positives (i.e., true padding positions correctly identified) and false positives (i.e., attack positions incorrectly identified as padding) are produced. Fig. 7 shows the figures for both PPI DiffKL and PPI KL. PPI DiffKL performs better in terms of FP, with a rate below 5% except for extremely short attacks. As far as TP are concerned, PPI DiffKL outperforms PPI KL for attacks of length approximately 25 or greater. We suspect that the reason for such a behaviour is related to the fact that PPI DiffKL makes used of both padding and attack information. While this certainly helps the algorithm to keep down the FP rate, it turns out to be a drawback when dealing with blocks when the attack portion is very short.

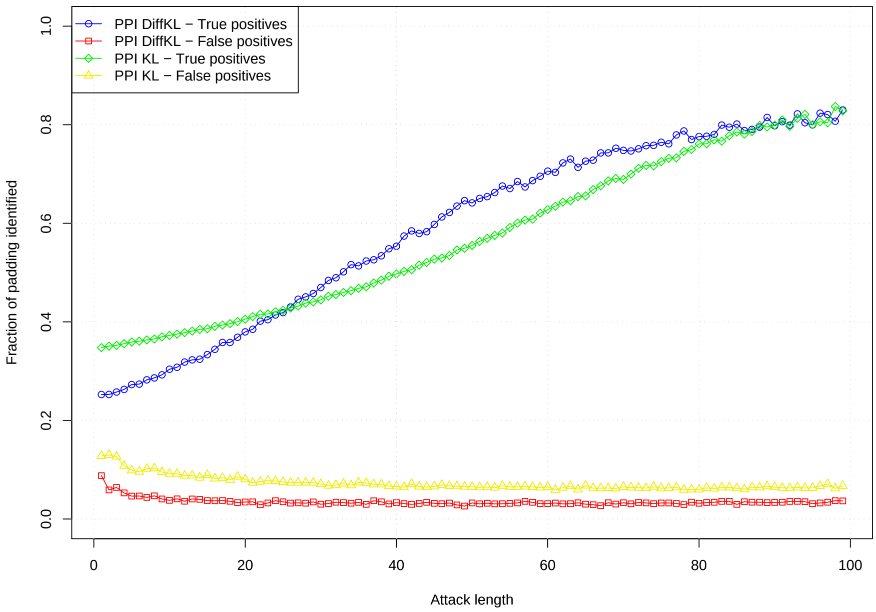

Figure 7: (In colour in the electronic version.) Accuracy of the PPI algorithm in identifying the padding portion of attacks of various lengths.

Regarding TP, the identification rate increases with the attack length almost linearly, up to a limit of around 80%. As we will see later, even these imperfect figures will be of help to assess the likelihood of an apparently normal block containing a mimicry attack.

The algorithm is reasonably fast. In our experiments the inclusion of the PPI increases the time required to process a block up to 11.717 ± 0.28 ms. Even though this is an increase of an order of magnitude compared with the time required by OCNB and OCNB-MRB, in a real-world system these figures do not constitute a problem, especially when considering that the analysis is performed every 100 user actions.

7. Masquerade Mimicry Attack Detection

In this section we describe how the PPI algorithm can be integrated within an anomaly detector to improve the identification of mimicry attacks. Even though we will limit our discussion to the case of OCNB, the same principle could be extended to a wider family of detectors.

In a first experiment, we generated 10000 blocks B containing mimicry attacks and applied the PPI algorithm to each one of them. We then have com-

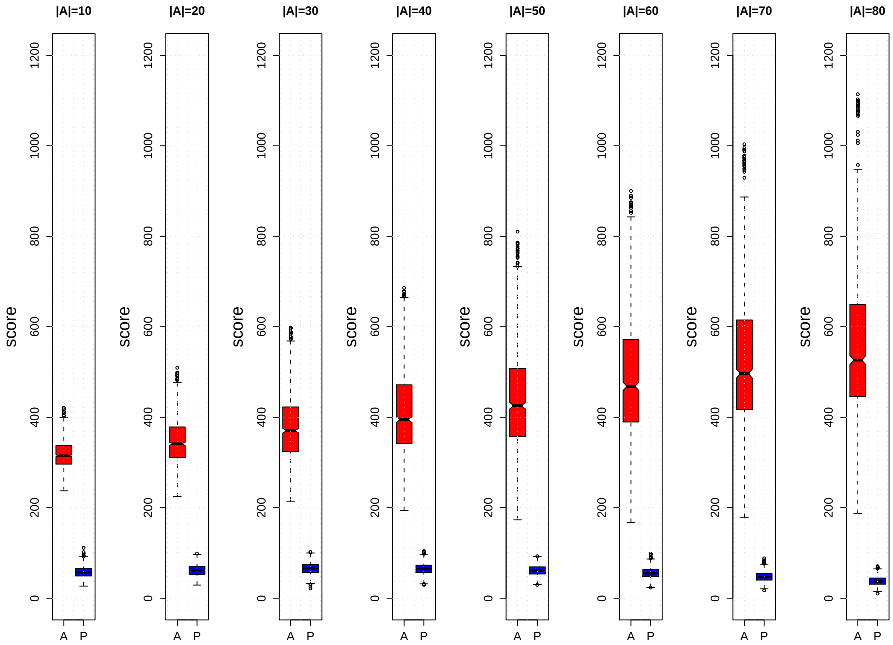

Figure 8: (In colour in the electronic version.) Score distribution for attack (red) and padding (blue) sections in blocks containing attacks of various lengths.

puted the anomaly score, given by (4), to each one of 2 sequences (attack and padding) returned by the algorithm *separately*. The purpose of this is to measure the contribution towards the overall anomaly score of the identified padding and attack portions. (Recall that the overall score is merely the sum of these two scores.) Fig. 8 shows the distribution of anomaly scores for the attack and padding sections for attacks of various lengths. As expected, padding sequences map to very low scores (around 50) which, besides, are almost independent of the attack length. On the contrary, the attack portion generally receives a much higher score, which obviously increases with the attack length.

When applied to self blocks, the result is completely similar. Nevertheless, in this case the identified "attack" portions correspond to false negatives of the PPI algorithm. These, however, are comparatively very few, a fact that will facilitate the construction of a combined anomaly score capable of detecting mimicry attacks. The measure we propose below is not the only way of exploiting this behaviour, but in our experiments it turned out to be the best performing. The idea consists of reusing the OCNB-based anomaly score and applying it to each portion, attack and padding, separately. The overall score is then computed as a weighted combination of both scores, with a major reward put on the attack portion:

$$score(B) = \begin{aligned} & -\sum_{c_i \in P} n_i(P) \log P(c_i|self) \\ & -\beta\Big(\sum_{c_i \in A} n_i(A) \log P(c_i|self)\Big) \end{aligned} \tag{9}$$

with $\beta \geq 1$. The effect of parameter β is clear and its value should be investigated empirically. In our experimentation (reported below), we found reasonable results for most users with values of β ranging between 2 and 8.

7.1. Experimental Results

Table 3 summarises the behaviour of the OCNB detector based on the use of expression (9). As before, each threshold has been tuned so as to limit the false positive rate to 5%. The first column (1v49) shows the detection rate computed as per the 1v49 experiment (i.e., blocks belonging to other users are considered as masquerading attempts, but no mimicry attack is included). Note that using the PPI algorithm generally has some impact on the detection rate of non-mimicry attacks. The reasons for this behaviour are related to the false positives generated by the identification algorithm, particularly in the case of users with similar profiles, as expression (9) tends to reduce the anomaly score of blocks coming from users with similar profiles. The overall effect, however, is very limited, and the global detection rate only degrades by less than 4% on average. The remaining columns in Table 3 show the fraction of detected mimicry attacks of lengths between 10 and 40. In all cases, the inclusion of the PPI algorithm increases the rate by more than 20%. For some users the improvement is enormous; see, for example, users 8, 16, 33, 34, or 49. In other cases (e.g., users 20, 26, 35) the algorithm is of little help. We have not investigated yet the reasons for this behaviour.

In general terms, the PPI-based detector achieves much better detection rates of mimicry attacks than OCNB with multiple random bags. As mentioned before, the process is indeed slower, but the sort of times here involved do not mean any problem for a real-world application. On the downside, the detection rate of non-mimicry attacks is slightly affected for some users. We expect to address this issue in future work.

8. Conclusions and Future Work

The majority of current approaches to identifying masquerade attempts ultimately rely on an anomaly detection algorithm and, consequently, are susceptible to evasion by a resourceful adversary. In this paper we have introduced the concept of mimicry attacks in the context of masquerade detection and given practical schemes to generate such attacks in the case of a widely used algorithm – the OCNB. From an adversarial point of view, the cost of generating a masquerade mimicry attack is negligible, and our experimental results show that most of these attacks can effectively evade detection.

Table 3: Detection rates (FP rate 5%) using the original OCNB (normal face) and the PPI-based OCNB (bold face).

User	1v49	$\lvert A\rvert = 10$	$\lvert A\rvert = 20$	$\lvert A\rvert = 30$	$\lvert A\rvert = 40$	β
0	0.805 / **0.653**	0.000 / **0.110**	0.070 / **0.368**	0.237 / **0.554**	0.359 / **0.643**	4.0
1	0.964 / **0.945**	0.392 / **0.970**	0.821 / **0.968**	0.937 / **0.974**	0.970 / **0.984**	4.0
2	0.968 / **0.958**	0.000 / **0.180**	0.080 / **0.676**	0.311 / **0.914**	0.573 / **0.946**	3.0
3	0.926 / **0.851**	0.039 / **0.401**	0.348 / **0.677**	0.576 / **0.790**	0.653 / **0.849**	4.0
4	0.806 / **0.805**	0.089 / **0.149**	0.426 / **0.467**	0.599 / **0.619**	0.687 / **0.674**	2.0
5	0.984 / **0.961**	0.018 / **0.150**	0.599 / **0.650**	0.872 / **0.897**	0.945 / **0.984**	4.0
6	0.819 / **0.706**	0.028 / **0.267**	0.292 / **0.526**	0.434 / **0.648**	0.550 / **0.692**	3.0
7	0.908 / **0.908**	0.000 / **0.002**	0.000 / **0.129**	0.005 / **0.438**	0.159 / **0.605**	5.0
8	0.767 / **0.668**	0.000 / **0.357**	0.191 / **0.574**	0.374 / **0.703**	0.460 / **0.709**	4.0
9	0.143 / **0.162**	0.000 / **0.000**	0.000 / **0.000**	0.000 / **0.010**	0.000 / **0.011**	4.0
10	0.780 / **0.647**	0.004 / **0.180**	0.214 / **0.457**	0.394 / **0.553**	0.508 / **0.613**	3.0
11	0.524 / **0.505**	0.000 / **0.085**	0.015 / **0.275**	0.080 / **0.387**	0.205 / **0.460**	4.0
12	0.059 / **0.053**	0.000 / **0.000**	0.000 / **0.000**	0.000 / **0.000**	0.000 / **0.000**	5.0
13	0.888 / **0.780**	0.002 / **0.374**	0.221 / **0.639**	0.461 / **0.775**	0.584 / **0.844**	4.0
14	0.716 / **0.625**	0.005 / **0.172**	0.186 / **0.392**	0.365 / **0.547**	0.465 / **0.600**	3.0
15	0.236 / **0.253**	0.000 / **0.089**	0.000 / **0.257**	0.000 / **0.429**	0.000 / **0.548**	6.0
16	0.924 / **0.875**	0.071 / **0.791**	0.319 / **0.896**	0.508 / **0.935**	0.668 / **0.933**	3.0
17	0.935 / **0.913**	0.000 / **0.000**	0.000 / **0.024**	0.000 / **0.231**	0.064 / **0.427**	6.0
18	0.851 / **0.795**	0.309 / **0.522**	0.560 / **0.687**	0.638 / **0.754**	0.712 / **0.781**	3.0
19	0.031 / **0.041**	0.000 / **0.085**	0.000 / **0.219**	0.000 / **0.282**	0.000 / **0.391**	8.0
20	0.904 / **0.886**	0.000 / **0.000**	0.000 / **0.000**	0.000 / **0.001**	0.096 / **0.132**	2.0
21	0.788 / **0.739**	0.128 / **0.670**	0.410 / **0.774**	0.557 / **0.781**	0.556 / **0.807**	3.0
22	0.942 / **0.901**	0.026 / **0.087**	0.253 / **0.464**	0.441 / **0.609**	0.627 / **0.742**	2.0
23	0.875 / **0.832**	0.008 / **0.375**	0.270 / **0.682**	0.501 / **0.774**	0.619 / **0.812**	3.0
24	0.861 / **0.816**	0.000 / **0.092**	0.078 / **0.431**	0.297 / **0.606**	0.503 / **0.657**	3.0
25	0.860 / **0.812**	0.000 / **0.015**	0.003 / **0.200**	0.085 / **0.488**	0.418 / **0.637**	3.0
26	0.016 / **0.004**	0.000 / **0.002**	0.000 / **0.012**	0.000 / **0.046**	0.000 / **0.116**	8.0
27	0.812 / **0.716**	0.000 / **0.262**	0.155 / **0.529**	0.377 / **0.646**	0.507 / **0.676**	4.0
28	0.251 / **0.209**	0.000 / **0.000**	0.000 / **0.001**	0.000 / **0.001**	0.000 / **0.014**	3.0
29	1.000 / **1.000**	1.000 / **1.000**	1.000 / **1.000**	1.000 / **1.000**	1.000 / **1.000**	1.0
30	0.837 / **0.787**	0.055 / **0.097**	0.390 / **0.394**	0.538 / **0.612**	0.679 / **0.712**	2.0
31	0.993 / **0.985**	0.844 / **0.996**	0.976 / **0.997**	0.988 / **0.999**	0.987 / **1.000**	4.0
32	0.764 / **0.725**	0.000 / **0.002**	0.000 / **0.036**	0.000 / **0.266**	0.014 / **0.544**	4.0
33	0.821 / **0.764**	0.101 / **0.642**	0.400 / **0.786**	0.585 / **0.817**	0.652 / **0.835**	4.0
34	0.971 / **0.931**	0.007 / **0.543**	0.643 / **0.892**	0.864 / **0.954**	0.903 / **0.960**	4.0
35	0.772 / **0.761**	0.000 / **0.000**	0.000 / **0.001**	0.000 / **0.001**	0.000 / **0.035**	2.0
36	0.773 / **0.785**	0.000 / **0.053**	0.127 / **0.380**	0.372 / **0.539**	0.460 / **0.638**	2.0
37	0.070 / **0.086**	0.000 / **0.097**	0.000 / **0.229**	0.000 / **0.370**	0.000 / **0.422**	9.0
38	0.033 / **0.043**	0.000 / **0.155**	0.000 / **0.269**	0.000 / **0.339**	0.000 / **0.378**	9.0
39	0.471 / **0.493**	0.000 / **0.089**	0.000 / **0.316**	0.000 / **0.489**	0.040 / **0.577**	5.0
40	0.510 / **0.566**	0.000 / **0.474**	0.000 / **0.669**	0.002 / **0.786**	0.051 / **0.809**	5.0
41	0.815 / **0.796**	0.000 / **0.000**	0.000 / **0.070**	0.054 / **0.243**	0.220 / **0.391**	2.0
42	0.460 / **0.426**	0.000 / **0.191**	0.000 / **0.438**	0.009 / **0.633**	0.066 / **0.725**	5.0
43	0.791 / **0.718**	0.000 / **0.095**	0.059 / **0.370**	0.218 / **0.593**	0.371 / **0.676**	3.0
44	0.649 / **0.602**	0.000 / **0.001**	0.003 / **0.042**	0.102 / **0.210**	0.289 / **0.352**	2.0
45	0.994 / **0.992**	0.908 / **0.926**	0.981 / **0.981**	0.989 / **0.988**	0.995 / **0.995**	2.0
46	0.991 / **0.986**	0.000 / **0.031**	0.000 / **0.535**	0.290 / **0.928**	0.786 / **0.982**	4.0
47	0.733 / **0.704**	0.005 / **0.073**	0.143 / **0.320**	0.311 / **0.477**	0.437 / **0.549**	3.0
48	0.598 / **0.576**	0.000 / **0.102**	0.000 / **0.329**	0.045 / **0.476**	0.157 / **0.572**	4.0
49	0.651 / **0.599**	0.000 / **0.661**	0.072 / **0.780**	0.284 / **0.802**	0.353 / **0.813**	4.0
Avg	0.701/ **0.667**	0.081 / **0.253**	0.206 / **0.423**	0.314 / **0.558**	0.407 / **0.625**	–

We have first studied the impact of randomising the detection procedure by using the MRB variant of OCNB. Our empirical analysis indicates that this scheme constitutes a detection strategy considerably more accurate than OCNB alone. Moreover, introducing a probabilistic component in the detection procedure does not seem to have an adverse impact on the detection quality of standard, non-mimicry masquerade attacks.

In order to improve upon the results exhibited by OCNB-MRB, we have proposed the PPI algorithm, a very efficient procedure that attempts to separate the attack sequence from the padding in a behavioural pattern. The rationale behind the PPI algorithm is sound and relies on the intuitive idea that the attack and padding segments have different information content, a fact that can be measured, for example, through the KL divergence. When tested under the same conditions as the previous two approaches, our experimental results show that the PPI performs significantly better with almost no degradation in terms of false positives. Moreover, the principle behind the PPI algorithm is general and can be adapted to detectors other than OCNB.

In future work we will explore the extent to which other detectors are vulnerable to masquerade mimicry attacks. For instance, previous research has shown that detectors based on SVM perform quite well in the masquerade setting [46]. It remains to be seen if efficient procedures for generating mimicry attacks against SVM do exist and, if so, how algorithms similar to the PPI can be developed. More generally, we anticipate that future research in this area should consider the presence of a sophisticated adversary with full knowledge of the internal functioning of the deployed sensors. This will lead to more robust designs, capable of enduring attacks carefully crafted to evade detection.

References

[1] M. Bertacchini and P.I. Fierens. "Preliminary Results on Masquerader Detection using Compression-based Similarity Metrics". *Electronic Journal of SADIO* **7**(1), 2007.

[2] B. Biggio, G. Fumera, and F. Roli. "Adversarial Pattern Classification Using Multiple Classifiers and Randomisation". *Structutal, Syntactic, and Statistical Pattern Recognition*, LNCS **5342**:500–509, 2008.

[3] B.M. Bowen, M. Ben Salem, S. Hershkop, A.D. Keromytis, and S.J. Stolfo. "Designing Host and Network Sensors to Mitigate the Insider Threat". *IEEE Security & Privacy*, pp. 22–29, Nov/Dec 2009.

[4] D.D. Caputo, G.D. Stephens, and M.A. Maloof. "Detecting Insider Theft of Trade Secrets". *IEEE Security & Privacy*, pp. 14–21, Nov/Dec 2009.

[5] E. Chan-Tin, D. Feldman, N. Hopper, and Y. Kim. "The Frog-Boiling Attack: Limitations of Anomaly Detection for Secure Network Coordinate Systems". In *SecureComm 2009*.

[6] L. Chen and G. Dong. "Masquerader Detection using OCLEP: One-class Classification using Legth Statistics of Emerging Patterns". In *WAIMW 2006*, pp. 5.

[7] N. Delvi, P. Domingos, Mausam, S. Sanghai, and D. Verma. "Adversarial Classification". In *ACM KDD 2004*, pp 98–108.

[8] F.A. Durán, S.H. Conrad, G.N. Conrad, D.P. Duggan, and E.B. Held. "Building a System for Insider Security". *IEEE Security & Privacy*, pp. 30–38, Nov/Dec 2009.

[9] J.A. Endler. "An overview of the relationships between mimicry and crypsis", *Biological Journal of the Linnean Society*, **16**(1):25–31, 1981.

[10] J.M. Estevez-Tapiador, P. Garcia-Teodoro, and J.E. Diaz-Verdejo. "Stochastic Protocol Modeling for Anomaly-Based Network Intrusion Detection". In *IWIA 2003*, pp. 3-12.

[11] J.M. Estevez-Tapiador, P. Garcia-Teodoro, and J.E. Diaz-Verdejo. "Detection of Web-based Attacks through Markovian Protocol Parsing". In *ISCC 2005*, pp. 457–462.

[12] S. Evans, E. Eiland, S. Markham, J. Impson, and A. Laczo. "MDLcompress for Intrusion Detection: Signature Inference and Masquerade Attack". In *MILCOM 2007*, pp. 1–7.

[13] P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee. "Polymorphic Blending Attacks". In *15th USENIX Security Symposium*, 2006.

[14] P. Fogla and W. Lee. "Evading network anomaly detection systems: formal reasoning and practical techniques". In *CCS 2006*, pp. 59–68.

[15] S. Forrest, S.A. Hofmeyr, A. Somayaji, and T.A. Longstaff. "A Sense of Self for Unix Processes". In *IEEE Symp. Security and Privacy*, 1996.

[16] S. Forrest, A.S. Perelson, L. Allen, and R. Cherukuri. "Self-Nonself Discrimination in a Computer". In *IEEE Symp. Security and Privacy*, 1994.

[17] D. Gao, M.K. Reiter, and D. Song. "On Gray-Box Program Tracking for Anomaly Detection". In *USENIX Security Symposium*, 2004.

[18] M. Gebski and R.K. Wong. "Intrusion Detection via Analysis and Modelling of User Commands". In *DAWAK*, LNCS Vol. 3589, pp. 388–397. Springer-Verlag, 2005.

[19] J.T. Giffin, S. Jha, and B.P. Miller. "Automated Discovery of Mimicry Attacks". In *RAID 2006*.

[20] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd Edition. Springer-Verlag, 2009.

[21] S. Hofmeyr, S. Forrest, and A. Somayaji. "Intrusion Detection Using Sequences of System Calls". *J. Computer Security* **6**:151–180, 1998.

[22] M.C. Jason Program Office. "Horizontal Integration: Broader Access Models for Realizing Information Dominance". *Technical Report JSR-04-132*, The MITRE Corporation, JASON Program Office, Mclean, Virginia, Dec 2004. `http://www.fas.org/irp/agency/dod/jason/classpol.pdf`.

[23] H.G. Kayacik, A.N. Zincir-Heywood, and M.I. Heywood. "Automatically Evading IDS Using GP Authored Attacks". In *IEEE Conf. on Computational Intelligence for Security and Defense Applications*, 2007.

[24] M. Kearns and M. Li. "Learning in the Presence of Malicious Errors". In *Proc. ACM Symposium on Theory of Computing*, pp 267–280, 1988.

[25] K.S. Killourhy and R.A. Maxion. "Toward Realistic and Artifact-Free Insider-Threat Data". In *ACSAC 2007*, pp. 87–96.

[26] C. Kruegel, E. Kirda, D. Mutz, W. Robertson, and G. Vigna. "Automating Mimicry Attacks using Static Binary Analysis". In *USENIX Security Symposium*, 2005.

[27] C. Kruegel, T. Toth, and E. Kirda. "Service Specific Anomaly Detection for Network Intrusion Detection". In *SAC 2002*, pp. 201–208.

[28] M. Latendresse. "Masquerade Detection via Customized Grammars". In *DIMVA 2005*, LNCS Vol. 3548, pp. 141–159. Springer-Verlag, 2005.

[29] D. Lowd and C. Meek. "Adversarial Learning". In *ACM KDD 2005*.

[30] M. Mahoney. "Network Traffic Anomaly Detection Based on Packet Bytes". In *Proc. ACM SAC*, 2003.

[31] M. Mahoney and P.K. Chan. "Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks". In *Proc. SIGKDD*, 2002.

[32] R.A. Maxion and T.N. Townsend. "Masquerade Detection using Truncated Command Lines". In *DSN 2002)*, pp. 219–228.

[33] R.A. Maxion. "Masquerade Detection using Enriched Command Lines". In *DSN 2003)*, pp. 5–14.

[34] M. Oka, Y. Oyama, H. Abe and K. Kato. "Anomaly Detection Using Layered Networks Based on Eigen Co-occurrence Matrix ". In *RAID 2004*, LNCS Vol. 3224, pp. 223–237. Springer-Verlag, 2004.

[35] S.L. Pfleeger and S.J. Stolfo. "Addressing the Insider Threat". *IEEE Security & Privacy*, pp. 10–13, Nov/Dec 2009.

[36] R. Posadas, J.C. Mex-Perera, R. Monroy, J.A. Nolazco-Flores. "Hybrid Method for Detecting Masqueraders using Session Folding and Hidden Markov Models". In *Proc. 5th Mexican Intl. Conf. on Artificial Intelligence*, pp. 622–631, 2006.

[37] M. Ben Salem, S. Hershkop, S. Stolfo. "A Survey of Insider Attack Detection Research". In *Insider Attack and Cyber Security: Beyond the Hacker*, Springer, 2008.

[38] M. Ben Salem and S. Stolfo. "Masquerade Attack Detection Using a Search-Behavior Modeling Approach". Columbia University, Computer Science Department, Technical Report CUCS-027-09, 2009.

[39] M. Schonlau, W. DuMouchel, W.-H. Ju, A.F. Karr, M. Theus, and Y. Vardi. "Computer Intrusion: Detecting Masquerades". *Statistical Science* **16**(1):58–74, Feb 2001.

[40] R. Sommer and V. Paxson. "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection". In *IEEE Symposium on Security and Privacy*, 2010.

[41] K.M.C. Tan, K.S. Killourhy and R.A. Maxion. "Undermining an Anomaly-Based Intrusion Detection Systems Using Common Exploits". In *RAID 2002*.

[42] K Tan, J. McHugh, and K.S. Killourhy. "Hiding Intrusions: From the Abnormal to the Normal and Beyond". In *Proc. 5th Information Hiding Workshop*, 2002.

[43] J.E. Tapiador and J.A. Clark. "Information-Theoretic Detection of Mimicry Masquerade Attacks". In *NSS 2010*, pp. 5–13.

[44] D. Wagner and R. Dean. "Intrusion detection via static analysis". In *Proc. of the 2001 IEEE Symposium on Security and Privacy*, pp.156-168, 2001.

[45] D. Wagner and P. Soto. "Mimicry Attacks on Host-Based Intrusion Detection Systems". In *ACM CCS 2002*.

[46] K. Wang and S. Stolfo. "One-class Training for Masquerade Detection". In *ICDM Workshop on Data Mining for Computer Security*, 2003.

[47] K. Wang and S. Stolfo. "Anomalous Payload-based Network Intrusion Detection". In *RAID 2004*.

[48] K. Wang and S. Stolfo. "Anomalous Payload-based Worm Detection and Signature Generation". In *RAID 2005*.

[49] C. Warrender, S. Forrest, and B. Pearlmutter. "Detecting Intrusions Using System Calls: Alternative Data Models". In *IEEE Symposium on Security and Privacy*, 1999.

[50] Y. Zhou, Z. Jorgensen, and M. Inge. "Combating Good Word Attacks on Statistical Spam Filters with Multiple Instance Learning." *IEEE ICTAI 2007*, pp. 298–305.