Article can be accessed online at http://www.publishingindia.com

Kullback-Leibler Divergence for Masquerade Detection

Geetha Ranjini Viswanathan*, Richard M. Low, Mark Stamp*****

Abstract

A masquerader is an attacker who gains access to a legitimate user's credentials and pretends to be that user so as to evade detection. Several statistical techniques have been applied to the masquerade detection problem, including hidden Markov models (HMM) and one class na¨ıve Bayes (OCNB). In addition, Kullback-Leibler (KL) divergence has been used in an effort to improve detection rates. In this paper, we analyze masquerade detection techniques that employ HMMs, OCNB, and KL divergence. Detailed statistical analysis is provided to compare the effectiveness of these various approaches.

Keywords: Masquerade Detection, Kullback-Leibler Divergence, One Class Naive Bayes, Hidden Markov Models, Intrusion Detection

1. Introduction

An intruder is an attacker who gains unauthorized access to a system. A masquerader is an intruder who carries out some malicious activity and attempts to avoid detection by pretending to be a legitimate user [24]. A masquerade detection system is an intrusion detection system that is specifically designed to detect such an intruder.

The research presented here represents an anomaly-based approach to masquerade detection. Specifically, we analyze UNIX commands for anomalous behavior. There is a vast amount of prior research on this particular problem; representative examples include [6], [8], [9], [10], [12], [17], [19], [24], [26], [27]. The survey paper [2] lists more than 40 relevant publications that appeared prior to 2009.

In this research, we use hidden Markov model (HMM) analysis, which can be viewed as a machine learning technique [27]. We train an HMM on each legitimate user's UNIX commands. These models are then used to determine the likelihood that a given set of commands is from the specified user or not. Prior research has shown that HMMs can be an effective technique for masquerade detection [10].

We also consider one class na¨ıve Bayes (OCNB). In OCNB, elementary statistical analysis is applied to a user's input data as a means to detect masqueraders. As with HMMs, OCNB has been successfully applied to the masquerade detection problem [10], [24].

Kullback-Leibler (KL) divergence is a statistical method that can be used to separate data from different distributions [11]. This technique has previously been studied in the context of masquerade detection based on OCNB [24]. From this previous work, it appears that KL divergence is a useful tool for improving detection rates. We have implemented and tested HMM-based and OCNB-based masquerade detection techniques. The contribution of this paper is a rigorous analysis of the effectiveness of KL divergence on detection rates for these two approaches. We show that using HMMs and KL divergence offers a significant improvement over the results obtained in previous research, where OCNB and KL divergence were used.

* Department of Computer Science, San Jose State University, San Jose, California, USA.
** Department of Mathematics, San Jose State University, San Jose, California, USA.
*** Department of Computer Science, San Jose State University, San Jose, California, USA. E-mail: stamp@cs.sjsu.edu

Section II introduces intrusion detection and the performance measures used in this project. Section III discusses relevant background information on HMMs, OCNB, and KL divergence. Section IV contains various test results. Finally, Section V concludes the paper and provides some possible directions for future work.

2. Background

In this section, we provide brief introductions to several relevant background topics. Specifically, we provide a brief introduction to intrusion detection before we turn our attention to the dataset used in this research and our performance measure.

2.1. Intrusion Detection

An intrusion is said to occur when an attacker gains access to a system. The goal of an intrusion detection system (IDS) is to identify whether current activity is legitimate or part of attack. IDS research has employed hidden Markov models [6], [8], [10], [27], one class na¨ıve Bayes [10], [19], [24], [26], and support vector machines [9], among many, many other techniques. In broad terms, we can classify an IDS as signature-based or anomaly-based [23].

An attack signature is a fixed pattern that represents a known attack [23]. Signature-based intrusion detection is relatively simple, efficient, and effective against attacks that have been previously observed and analyzed. The weakness of such an approach is that the attack must be known in advance and a useful signature must have been extracted. Consequently, previously unknown attacks are unlikely to be detected.

A anomaly-based IDS relies on the assumption that there will be a measurable difference between attacks and legitimate use [14]. An anomaly-based IDS operates in two phases, training and detection [7]. In masquerade detection, the goal of the training phase is to determine a model that fits a given user's behavior. Then during the detection phase, we use the model to detect significant deviations from the user's expected behavior.

Unlike signature-based systems, anomaly-based systems have the potential to detect previously unknown attacks. However, the false positive rate of an anomaly based system is likely to be much higher than a signature based system, and the training phase may be complex and costly [7].

2.2. Schonlau Dataset

The Schonlau dataset [18] consists of truncated user issued UNIX commands. The data file contain 15.000 UNIX commands for each of 50 users, where each user's data is composed of 150 blocks of 100-commands each. For each user, the first 50 blocks (i.e., 5000 commands) are training data, while the remaining 100 blocks (i.e., 10.000 commands) are the test set. The training set consists entirely of commands generated by the specified user, while some blocks in the test set are user commands and other blocks are "attack" commands—the attacks are simply blocks that have been selected from other user profiles.

The dataset also includes a map file, in the form of a 0-1 matrix of size 100×50, which indicates which of the test blocks are attacks and which are not. In this matrix, a 0 represents that the corresponding user test block is attack-free while a 1 represents that the block contains attack data. The attack-free blocks (i.e., the blocks that belong to the specified user) are referred to as "self" blocks and the attack blocks are "non-self" blocks [24].

2.3. Performance Measures

Any attempt at detection can yield one of four results. If the data is an attack and it is identified as such, we have a true positive. If the data is not an attack but it is identified as an attack, then a false positive has occurred. If an attack is not identified as such, we have a false negative, and, finally, if legitimate (attack-free) data is identified as legitimate, it is a true negative. These four outcomes are illustrated in the form of a confusion matrix in Figure 1.

Figure 1: Confusion Matrix

		Predicted Class	
		P	N
Actual Class	P	TP	FN
	N	FP	TN

Performance measures such as the true positive rate (TPR), false positive rate (FPR), and accuracy can be used to determine the effectiveness of a detection system. These measures are computed as

$$\begin{aligned} \text{TPR} &= \frac{\text{TP}}{\text{TP}+\text{FN}} \\ \text{FPR} &= \frac{\text{FP}}{\text{FP}+\text{TN}} \\ \text{Accuracy} &= \frac{\text{TP}+\text{TN}}{\text{TP}+\text{FN}+\text{FP}+\text{TN}} \end{aligned} \quad (1)$$

Receiver operating characteristic (ROC) curves provide us with a means for comparing the effectiveness of detection strategies. For the research presented here, ROC curves are obtained by plotting the FPR on the *x*-axis versus the TPR on the *y*-axis, where the threshold is varied over the range of scores [4]. Once the ROC plots are obtained, we can then compute the area under the ROC curve (AUC). Ideal detection will result in an AUC of 1, while classification based on coin flipping will yield an AUC of approximately 0.5. The AUC is useful, since it provides a single, meaningful number that we can use to compare the performance of different detection strategies [3].

3. Detection Techniques

In this section, we discuss the various techniques that are employed in the masquerade detection research considered in this paper. Specifically, we discuss hidden Markov models, one class na¨ıve Bayes, Kullback-Leibler divergence, and scoring methods based on these techniques.

3.1. Hidden Markov Model

A hidden Markov model (HMM) is a machine-learning technique that can be used to build a model based on a given sequence of input data. HMMs have been successfully applied to a wide variety of problems, including malware detection [16], [21]. In addition, HMMs have been extensively studied in the context of anomaly-based intrusion detection. Previous research has shown that when applying HMMs to the Schonlau dataset, attacks of length 30 or more are effectively detected [10].

An HMM is denoted as $\lambda = (A, B, \pi)$, where A is the state transition matrix for the underlying Markov process, B is the observation probability distribution matrix, and π is the initial state distribution matrix. All three of these matrices are row-stochastic, that is, each row meets the requirements of a discrete probability distribution.

The following notation is commonly used for HMMs [22]:

T = length of the observation sequence

N = number of states in the Markov model

M = number of unique observation symbols

$Q = \{q_0; q_1; : : : ; qN - 1\}$ = the unique states

$V = \{0; 1; : : : ; M - 1\}$ = set of possible observations

$\varphi = (\varphi_0; \varphi_1; : : : ; \varphi_T - 1)$ = observation sequence

Figure 2 illustrates the concept behind an HMM. Note that the hidden states of the underlying Markov process are represented by $X_0; X_1; X_2; : : : ; X_{T-1}$. The A matrix drives the Markov process, while the B matrix probabilistically relates the hidden states to the observations. The region above the dashed line represents the "hidden" part of the HMM. For more details on HMMs, see, for example [22].

For the research presented in this paper, we train 50 HMMs, one for each user in the Schonlau dataset. Using these trained models, we compute scores on selected test data. After obtaining these baseline results, we experiment with Kullback-Leibler divergence to determine how best to improve on the baseline results.

3.2. One Class Na¨ıve Bayes

One class na¨ıve Bayes (OCNB) is another learning algorithm that has been successfully applied to the masquerade detection problem. OCNB is a straightforward classifier derived from Bayes rule. Using OCNB, we can compute the probability that an instance x belongs to a class y. By using

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

where $x = (x_1; x_2; : : : ; x_m)$, we can determine the class y that maximizes the probability of a given observation x [24].

In this paper, OCNB is used to compare the probability of a command occurring in the test data to the probability of the same command occurring in a given user's training data. We use these results to classify whether a series of commands corresponds to an attack or not. Previous

research involving the Schonlau dataset has shown this technique to be effective at detecting attack sequences of length 50 or more [10].

Given a block B of 100 commands, we derive the vector $[n_1(B); n_2(B); : : : ; n_m(B)]$, where $n_i(B)$ is the number of times that command c_i appears in the block B and m is the number of distinct commands [24]. For OCNB, the probability that the block B belongs to user y is computed as

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

where, since the priors are unknown, we let $P(y) = 1$.

The values $P(c_i|y)$ in (2) are derived from the training set for user y according to

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

where $T(y)$ is the training set for user y (i.e., the self data). The parameter α is used to ensure that all commands have a non-zero probability. Following previous work [12], [26], here we have selected $\alpha = 0.01$.

Note that in (2), the value $n_i(B)$ corresponds the block B that is to be tested against the training set. In contrast, in (3), the value of $n_i(B)$ is derived from the training set.

We can use (2) to compute a score as [24]

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

The higher the score in (4), the more anomalous the test block and hence the more likely it represents an attack. On the other hand, a score closer to 0 is indicative of a self block.

3.3. Kullback-Leibler Divergence

For two discrete probability distributions P and Q, the Kullback-Leibler (KL) divergence of Q from P is given by [24]

$$D_{KL}(P||Q) = \sum P(i) \log \frac{P(i)}{Q(i)}$$
$$= -\sum_i P(i) \log Q(i) + \sum_i P(i) \log P(i)$$

KL divergence provides a means for distinguishing observations from different distributions [11]. KL divergence has been used for identifying anomalies in wireless signals [1], biomedical data [15], and network traffic [5], [13], among many other applications. In [24], KL divergence is applied in an attempt to improve on OCNB-based masquerade detection. Next, we consider the technique in [24] in more detail.

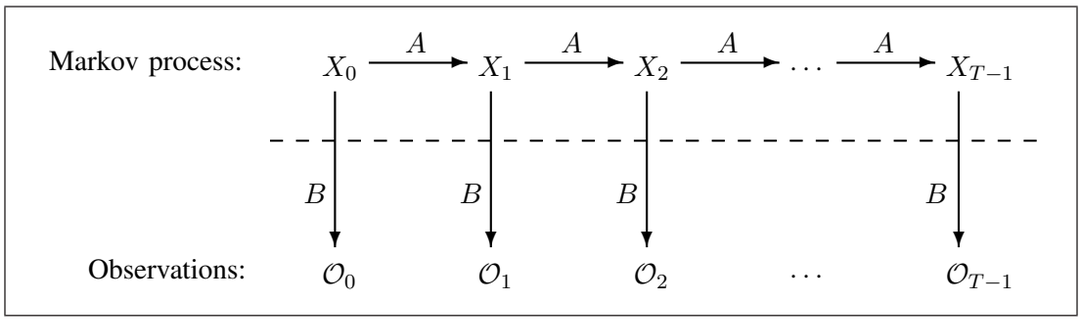
In [24], a probabilistic padding identification (PPI) algorithm is used to distinguish padding commands from attack commands. This algorithm makes use of the KL divergence.

Let A and P represent the attack and padding portions, respectively, for a given block B. Let M represent the trained model corresponding to the padding P. Then, in general, P should match M more closely than A matches M.

In the PPI algorithm, we attempt to find subsets $\hat{P}; \hat{A} \subseteq B$, with $\hat{P} \cup \hat{A} = B$ and $\hat{P} \cap \hat{A} = \phi$, such that $D_{KL}(\hat{P}||M)$ is "low" and $D_{KL}(\hat{A}||M)$ is "high". That is, we want to partition B into P and A so that P closely matches M, while A does not.

For any putative choice of A and P, we compute a score for the partition as

Figure 2. Hidden Markov process [22]



$$|D_p - D_a| = \left| \sum_i |P| \log \frac{|\hat{P}|}{|M|} - \sum_i |\hat{A}| \log \frac{|\hat{A}|}{|M|} \right|$$

where the notation $|X|$ represents the number of elements in the set X.

We have implemented and analyzed two versions of the PPI algorithm, namely, an attack-first and a padding-first version. These names refer to the initialization process—in the attack-first case, we initialize $A = B$ and $P = \phi$, while in the padding-first case, we initialize $A = \phi$ and $P = B$. The attack-first strategy is analyzed in [24], while the padding-first approach seems like a logical strategy. We have implemented both and we compare the results for HMM and OCNB based detectors in Section IV.

As given in [24], for the attack-first PPI we determine A and P using Algorithm 1 which, in turn, uses Algorithm 2. For the padding-first PPI we make the obvious modification; see Algorithm 3.

Finally, given A and P, we compute a PPI score as

$$\text{score}_{PPI}(B) = -\text{score}(P) - \beta.\ \text{score}(A) \quad (5)$$

where "score" is defined in (4) and $\beta \geq 1$ is a weight. Note that in [24], different values of β are selected for different users so as to optimize detection rates. Here, for simplicity we use a fixed value of $\beta = 2$ for all users.

3.4. Masquerade Attacks

As in [24], we assume that the attacker has perfect knowledge (including all details of the detection algorithm), the detector is not poisoned (i.e., the detection system is trained on attack-free data), and the attack sequence is contained within a single 100-command block. Also, we assume that each attack command can be placed at any location within the block.

Figure 3 illustrates an example of a masquerade,

Algorithm 1 Attack-first PPI

Input: Block B, Model M

Output: Boolean vector $C(i)$ = true if $B(i)$ is padding

1: Initially $C(i) \leftarrow$ false for all i
2: for $i = 1$ to $|B|$ do
3: $\bar{d} = \text{DiffKL}(C, B, M)$
4: $C(i) \leftarrow$ true
5: $\bar{d} = \text{DiffKL}(C, B, M)$
6: if $(d \leq d)$ then
7: $C(i) \leftarrow$ false
8: end if
9: end for
10: return P = commands $B(i)$ such that $C(i)$ is true

Algorithm 2 DiffKL

Input: Boolean vector C, block B, model M

Output: Difference of KL divergences

1: $\hat{A} \leftarrow$ PDF of those $B(i)$ such that $C(i)$ is false
2: $\hat{P} \leftarrow$ PDF of those $B(i)$ such that $C(i)$ is true
3: $D_a \leftarrow D_{KL}(\hat{A}\,|M|)$
4: $D_p \leftarrow D_{KL}(\hat{P}\,|M|)$
5: return $|D_p - D_a|$

Algorithm 3 Padding-first PPI

Input: Block B, Model M

Output: Boolean vector $C(i)$ = true if $B(i)$ is padding

1: Initially $C(i) \leftarrow$ false for all i
2: for all $B(i)$ in M, $C(i) \leftarrow$ true
3: $d = \text{DiffKL}(C, B, M)$
4: for $i = 1$ to $|B|$ do
5: $C(i) \leftarrow$ false
6: $\bar{d} = \text{DiffKL}(C, B, M)$
7: if $(d < \bar{d})$ then
8: $C(i)$ $\bar{\text{d}}$ true
9: end if
10: end for
11: return P = commands $B(i)$ such that $C(i)$ is true

or mimicry, attack. The boxed commands represent the attack and the unboxed commands are the padding.

Figure 3. Example of Mimicry Attack [24]

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

To test the performance of the HMM and OCNB-based systems analyzed here, masquerade attacks were generated for each user. We have used commands from other users as attack data. For each user, 50 attack blocks were generated for each of the attack lengths 10, 20, 30, . . . ; 100, with the attack commands randomly distributed throughout the block. Therefore, the performance of each technique was tested on a total of 500 attack blocks.

4. Experimental Results

In this section, we present detailed test results for the performance of our HMM and OCNB based masquerade detection systems (MDS). We have performed tests using both the HMM and OCNB for each of the following three cases:

- No probabilistic padding identification (PPI)
- Padding-first PPI
- Attack-first PPI

In addition, for the PPI tests, we experimented with different scoring methods. For all experiments, our performance measure is the area under the curve (AUC) of ROC curves, as discussed in Section 2.3. For the sake of brevity, we only include selected results in this paper; additional experimental results can be found in [25].

We generate attack blocks as described in Section 3.4. That is, for each user 50 attack blocks were generated for each attack length

$$|A| \in \{10, 20, 30, \ldots, 100\}.$$

Recall that all blocks B are of length 100, and hence for each experiment, the padding P satisfies $|P| = 100 - |A|$.

To test the performance in the "no PPI" case, the attack blocks were directly scored against the appropriate trained model (HMM and OCNB). These results provide a baseline against which we can compare the performance of the PPI-based approaches. Recall that the PPI techniques employ Kullback-Leibler divergence.

To analyze the performance of padding-first and attack-first PPI, the attack blocks were first put through the appropriate PPI, which yields putative attack and padding subsequences for each block. These sequences were then scored using the score in (5).

Before we present our MDS results, we first consider the success of the PPI approaches at separating the padding and attack commands from a given masquerade attack block. These results appear in the next section.

4.1. PPI Comparison

The graph in Figure 4 provides a direct comparison of the accuracy of padding-first PPI and attack-first PPI. These results are based on the identification of attack data and padding data in blocks, with accuracy determined using the formula in (1). In the Schonlau dataset, it is known that some users' behavior is easier to model than others—the "hard," "easy," and "intermediate" cases in Figure 4 represent users that have proven hard, easy, and average to model. Specifically, users 47, 49, and 1 were used for the hard, easy, and intermediate cases, respectively, with the attacks selected from users 2 and 3 data in each case. Also, 100 attacks were tested for each case (easy, hard, intermediate) for each attack of length 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100.

Figure 4. Accuracy of Padding-first and Attack-first PPI

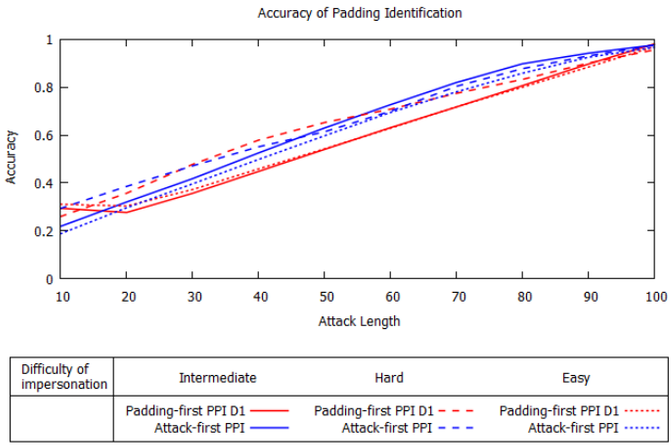
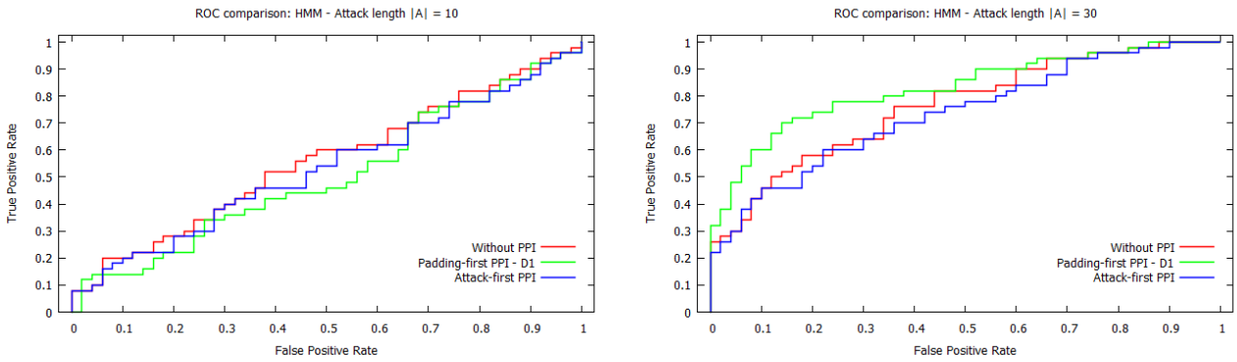


Figure 5. HMM: ROC Comparisons



(a) $|A| = 10$ (b) $|A| = 30$

From Figure 4 we see that neither the attack-first PPI nor the padding-first PPI appears to have a significant advantage. However, below we see that when scoring blocks, the padding-first PPI generally outperforms the attack-first PPI by a significant margin.

4.2. HMM Results

In this section we consider an HMM-based MDS. We compare results obtained when no PPI is used to those obtained using the padding-first PPI and attack-first PPI.

We first score the attack blocks for each user and each attack length using HMMs trained for the appropriate user. The resulting scores are compared to scores obtained on self blocks.

These experiments were then repeated with padding-first PPI applied to each masquerade attack block, with the scores computed using (5). Finally, we repeated this latter experiment using the attack-first PPI.

In Figure 5 we present ROC curves comparing these three different HMM-based approaches (no PPI, padding-first PPI and attack-first PPI). Figure 5 (a) gives results for attack length 10, while Figure 5 (b) gives the ROC comparison for attack length 30. These results indicate that none of the three methods succeeds at detecting attacks of length 10 at a rate significantly better than guessing. On the other hand, the results for attack length 30 show that padding-first PPI yields a slight improvement.

Figure 6 shows the AUC comparison for the no PPI, padding-first PPI and attack-first PPI cases for each attack length tested. From these results, we see that for the HMM-based detector under consideration, padding-first PPI shows improved detection rates for attacks of length about 25 or more. In contrast, the attack-first PPIbased

HMM detection rates are lower than the rates obtained using no PPI for all attack lengths. See [25] for additional HMM results.

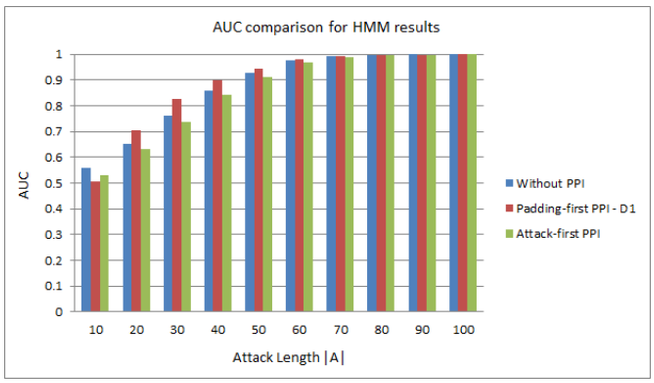
4.3. OCNB Results

In this section we consider an OCNB-based MDS. As in the previous section, we compare results obtained when no PPI is used with results for the padding-first PPI and attack-first PPI.

Figure 7 (a) shows the ROC comparison for attacks of length 10, while Figure 7 (b) gives results for attack length 30. As in the HMM case, for attack length 10 the detection rates are essentially equivalent to flipping a coin. However, for attacks of length 30, padding-first PPI achieves about a 50% improvement over the no PPI detection rate.

Figure 8 shows the AUC comparison for no PPI, padding-first PPI, and attack-first PPI using OCNB. For attack lengths of 30 or more, the padding-first PPI shows significant improvement in detection rate, whereas the detection rates of attack-first PPI for any attack length is lower than OCNB with no PPI. See [25] for additional OCNB results.

Figure 6. HMM: AUC Comparison



4.4. Discussion

The results in Figure 6 show that for our HMM-based MDS, the padding-first PPI generally offers a slight improvement over no PPI for attack lengths greater than 10. From Figure 8, we see that for our OCNB-based MDS, the padding-first PPI is again superior, and in this case, the improvement is more significant. For OCNB, the attack-first PPI performs poorly in all cases.

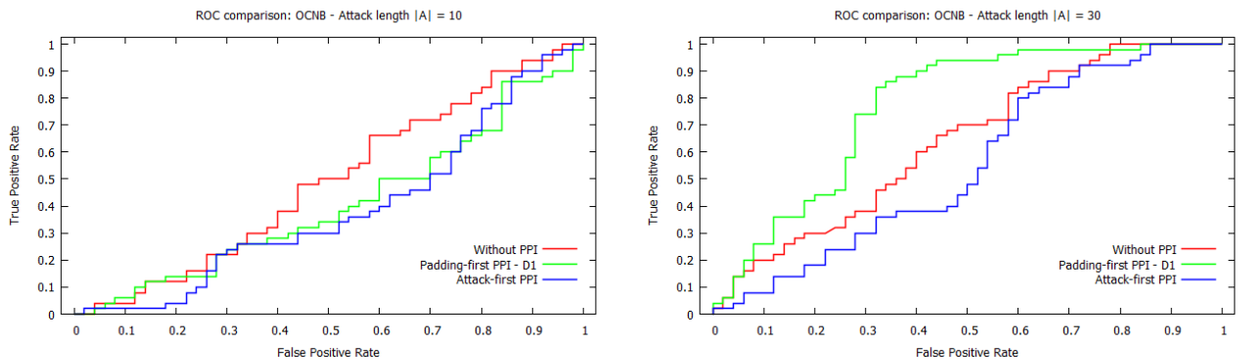
A comparison of Figures 6 and 8 shows that HMMs yield superior results to OCNB in every case. Consequently, the overall value of using PPI appears to be very limited, since an HMM-based detector with no PPI can achieve equivalent or better results than OCNB with PPI. In addition, PPI offers only a modest improvement when using HMMs. It is interesting that the authors of [24] chose to study PPI in the context of OCNB-based detection, where the relative improvement is large. However, in comparison to other more sophisticated strategies such as HMMs, the relative improvement of PPI appears to be much less pronounced. Nevertheless, even relatively small improvements in detection can be significant in the challenging field of masquerade detection.

5. Conclusions and Future Work

In this paper, we considered masquerade detection systems based on hidden Markov model (HMM) analysis and one class naïve Bayes (OCNB). For both, we experimented with two types of probabilistic padding-first identification (PPI), namely, attack-first and paddingfirst. The purpose of the PPI is to separate attack commands from padding commands, and thereby improve the scoring results. These PPI approaches both rely on Kullback-Leibler (KL) divergence. The attack-first PPI has been analyzed in previous research [24], while the padding-first PPI is appears to be new. Our experiments were conducted using the Schonlau dataset of UNIX commands, which has formed the basis for a large volume of published masquerade detection research.

For both PPI algorithms, we measured the detection rates of the HMM and OCNB-based masquerade detection systems for attack lengths ranging between 10% to 100% of the block length. Our results indicate that padding-first PPI is of significant value for OCNB-based detection, but of more limited value for HMM-based detection. In addition, our HMM-based detectors with no PPI generally performed better than OCNB-based detectors with PPI.

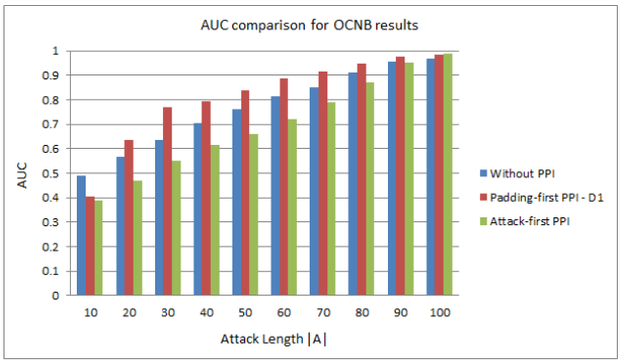
Figure 7. OCNB: ROC Comparisons



(a) $|A| = 10$ (b) $|A| = 30$

The research in this paper was limited to HMM and OCNB-based masquerade detection. This work could easily be extended to other related techniques, such

Figure 8. OCNB: AUC Comparison



as support vector machines [20]. The attack-first PPI and padding-first PPI algorithms considered here are the extreme cases. It is possible that better results would be obtained by an initialization scheme that includes an appropriate mixture of attack and padding commands.

In addition, a hill climb-based PPI strategy could be considered.

The test case considered here was UNIX commands, but the same techniques could be applied to other examples of user behavior-based data, such as keyboard dynamics, mouse movements, and so on. More generally, the techniques considered here could be applied to intrusion detection problems other than the masquerade detection problem considered here.

Finally, further research should be conducted to determine whether there are improved attacks that can better evade the systems analyzed here. If such attack strategies are found, those results could, in turn, be used to strengthen statistical-based masquerade detectors.

References

1. Afgani, M. (2008). Anomaly detection using the Kullback-Leibler divergence metric, *Applied Sciences on Biomedical and Communication Technologies. ISABEL' 08, First International Symposium*, 1-5.
2. Bertacchini, M. & Fierens, P. I. (2009). A Survey on Masquerader Detection Approaches, CIBSI 2009. Retrieved from www.criptored.upm.es/cibsi/cibsi2009/docs/Papers/CIBSI-Dia2-Sesion5(2).pdf
3. Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7), 1145-1159.
4. Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8), 861-874.
5. Gu, Y., McCallum, A. & Towsley, D. (2005). Detecting *Anomalies in Network Traffic Using Maximum Entropy Estimation*. IMC '05 Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement, (pp. 32-37).
6. Huang, L. & Stamp, M. (2011). Masquerade detection using profile hidden Markov models. Computers and Security, 30(8), 732-747.
7. Idika, N. & Mathur, A. (2007). A survey of malware detection techniques, Technical report, Software Engineering Research Center. Retrieved from www.serc.net/system/files/SERC-TR-286.pdf
8. Khanna, R. & Liu, H. (2008). Control theoretic approach to intrusion detection using a distributed hidden Markov model. *IEEE Wireless Communications*, 15(4), 24-33.
9. Kim, H. & Cha, S. (2005). Empirical evaluation of svm-based masquerade detection using unix commands. *Computers and Security*, 24(2), 160-168.
10. Kothari, A. (2012). Defeating Masquerade Detection, Master's Project 239. Retrieved from scholarworks.sjsu.edu/etd_projects/239
11. Kullback, S. & Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1), 79-86.
12. Maxion, R. A. & Townsend, T. N. (2004). Masquerade Detection Augmented with Error Detection. *IEEE Transactions on Reliability*, Special Section on Quality/Reliability Engineering of Information Systems, March, 53(1), 124-147.
13. Mun, G. J., Noh, B. N. & Kim, Y. M. (2009). Enhanced stochastic learning for feature selection in intrusion classification. International Journal of Innovative Computing, Information and Control, 5(11), 3625-3635.
14. Murali, A. & Rao, M. (2005). A survey on intrusion detection approaches. Information and Communication Technologies, ICICT 2005, 233-240.
15. Oh, J. H., Gao, J. & Rosenblatt, K. (2008). Biological data outlier detection based on Kullback-Leibler divergence. *Bioinformatics and Biomedicine, BIBM '08*, 24(16), 249-254.

16. Runwal, N., Low, R. M. & Stamp, M. (2012). Opcode graph similarity and metamorphic detection. *Journal in Computer Virology*, 8(1/2), 37-52.
17. Schonlau, M. & Theus, M. (2000). Detecting masquerades in intrusion detection based on unpopular commands. *Information Processing Letters*, 76(1/2), 33-38.
18. Schonlau, M. (2009). Masquerding User Data. Masquerade Data. Retrieved from www.schonlau.net/intrusion.html
19. Sharma, A. & Paliwal, K. (2007). Detecting masquerades using a combination of na¨ıve Bayes and weighted RBF approach. *Journal in Computer Virology*, 3(3), 237-245.
20. Shetty, S., Mukkavilli, S. K. & Keel, L. H. (2011). An Integrated Machine Learning and Control Theoretic Model for Mining Concept Drifting Data Streams. IEEE International Conference on Technologies for Homeland Security (HST).
21. Sridhara, S. M. & Stamp, M. (2013). Metamorphic worm that carries its own morphing engine. *Journal of Computer Virology and Hacking Techniques*, 9(2), 49-58.
22. Stamp, M. (2011). Information Security: Principles and Practice (2nd ed.). Wiley.
23. Stamp, M. (2012). A Revealing Introduction to Hidden Markov Models. Retrieved from www.cs.sjsu.edu/˜stamp/RUA/HMM.pdf
24. Tapiador, J. & Clark, J. (2011). Masquerade mimicry attack detection: A randomized approach. *Computers and Security*, 30(5), 297-310.
25. Viswanathan, G. R. (2013). Analysis of Kullback-Leibler Divergence for Masquerade Detection, Master's Project 302. Retrieved from scholarworks.sjsu.edu/etd_projects/302/
26. Wang, K. & Stolfo, S. (2003). One Class Training for Masquerade Detection. 3rd IEEE Conference Data Mining Workshop on Data Mining for Computer Security. Retrieved from cs.columbia.edu/˜kewang/paper/ DMSEC-camera.pdf
27. Yin, Q. (2003). Intrusion detection based on hidden Markov model. *Machine Learning and Cybernetics*, 5(1), 3115-3118.