Principled Reasoning and Practical Applications of Alert Fusion in Intrusion Detection Systems

Guofei Gu
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA
guofei@cc.gatech.edu

Alvaro A. Cárdenas
Department of EECS
University of California
Berkeley, CA 94720, USA
cardenas@cs.berkeley.edu

Wenke Lee
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA
wenke@cc.gatech.edu

ABSTRACT

It is generally believed that by combining several diverse intrusion detectors (i.e., forming an IDS ensemble), we may achieve better performance. However, there has been very little work on analyzing the effectiveness of an IDS ensemble. In this paper, we study the following problem: how to make a good fusion decision on the alerts from multiple detectors in order to improve the final performance. We propose a decision-theoretic alert fusion technique based on the likelihood ratio test (LRT). We report our experience from empirical studies, and formally analyze its practical interpretation based on ROC curve analysis. Through theoretical reasoning and experiments using multiple IDSs on several data sets, we show that our technique is more flexible and also outperforms other existing fusion techniques such as AND, OR, majority voting, and weighted voting.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Network**]: Security and Protection; B.8.2 [**PERFORMANCE AND RELIABILITY**]: Performance Analysis and Design Aids

General Terms

Security

Keywords

Intrusion detection, alert fusion, IDS ensemble, likelihood ratio test, ROC curve

1. INTRODUCTION

Diversity is a well-established approach for increasing the reliability of systems: if we want a computer service to be always available, we can use diversity to make it more difficult for an attack to take down an entire set of diverse computers or mechanisms. In Intrusion Detection Systems (IDSs), different detectors provide complementary information about the patterns to be classified. Some detectors[1] are effective in detecting a certain type of attack, while others aim to detect different types. A single intrusion detection algorithm (or anomaly classifier) cannot easily detect all kinds of intrusions/anomalies. Similarly, different classifiers are likely to make different kind of errors on normal data. Thus, a reasonable approach is not to rely on any single classifier for a decision, but a fusion rule to combine diverse classifiers to reach a final decision.

Now suppose we have a set of diverse IDSs reporting on the same event. If some IDSs output an alarm while others do not, should the fusion rule output an alarm? This is the so-called IDS fusion problem. A conservative (or, security-conscious) fusion rule will always output an alarm whenever one of the underlying IDSs outputs an alarm. However, this simple approach is inadequate in many situations. It does not filter the alerts from underlying IDSs and does not resolve conflicting votes from the IDSs (i.e., when some say "normal" and others say "attack" or "suspicious"). As a result, it produces as many alarms as the sum of all alarms from all the underlying IDSs. And since it does not provide a confidence attribute to the alarms, security analysts can be overwhelmed by the need to inspect a large number of alerts (including many false alarms).

In this paper, we make the following contributions:

- We propose the use of the likelihood ratio test (LRT) technique as a *principled* and *practical* way of combining different alarm reports.
- We study the effectiveness of the LRT rule empirically by using several IDSs on two different data sets and comparing its performance to several other fusion approaches.
- We provide a novel theoretical analysis of the LRT rule, and the resulting binary decision functions. We also show the practical intuition of the LRT rule when applied to the IDS fusion problem, and provide a novel practical interpretation of these results using ROC curves.
- We test the robustness of the LRT fusion rule with respect to possible parameter estimation errors (these parameters include the false positive rate, false negative rate, and base rate).

Our analysis shows that the LRT rule has several key characteristics:

ASIACCS '08, March 18-20, Tokyo, Japan

[1] Although there are some difference between the terms of detector, sensor, and classifier, they are basically the same components in our model (Figure 1). In the rest of the paper, we may use IDS/detector/sensor/classifier interchangeably.

- If the risk model is known (i.e., if we have the knowledge of the operating costs of the IDS, and the likelihood of an attack), the LRT rule can be shown to be the fusion rule that minimizes the average cost.
- Instead, if we do not assume any costs, we can use Neyman-Pearson theory to show that by tuning a threshold, we find the optimal tradeoff between false alarms and missed detections.
- Finally, if we do not make any assumption, the LRT fusion rule can be seen as a principled *ranking* algorithm, placing a level of suspicion on each event, depending on which IDSs output alarms (and which ones did not). A ranking algorithm will thus help the operators of IDSs identify and check only the most suspicious events (given their time constraints).

Our results also show that our approach outperforms other fusion approaches (e.g., AND, OR, majority voting, weighted voting) in achieving the lower overall cost in various risk scenarios, and in providing a principled ranking algorithm for alarms.

The paper is organized as follows. Section 2 reviews related work. In Section 3 we formalize the IDS fusion problem and describe our LRT-based alert fusion technique. In Section 4 we discuss the effectiveness of our approach by comparing it with other methods using several IDSs on two different data sets. In Section 5 we provide a novel interpretation of the LRT ensemble, and theoretically analyze why the LRT fusion rule is a better choice than other approaches. In Section 6 we discuss several extensions. Section 7 summarizes the paper and outlines further work.

2. RELATED WORK

There are several ways for combining classifiers in the machine learning literature. For ensemble methods [29, 14] such as bagging [8], or boosting [17], the goal is to generate different classifiers by resampling or reweighing the training data set. The decision rules applied to the generated classifiers are majority voting or weighted voting. In contrast, our method takes any *given* set of binary classifiers and finds optimal combination rules from a more general space of functions (i.e., our fusion rules are not restricted to be majority voting or weighted voting).

Another method for combining classifiers is stacking [51]. Stacking trains a meta-learner to combine the predictions of several base classifiers. This meta-learner is in general, a machine learning algorithm such as a neural network, or a support vector machine. The problem with these algorithms is that their classification rules are difficult to interpret, and that their objective is to minimize the probability of error; a metric that is not well suited for the evaluation of intrusion detection systems [9, 30, 20]. Our method, however, can be considered to be a stacking approach with the LRT rule as a particular meta-classifier. We have showed that LRT performs well and can outperform previously proposed stacking approaches [5]. In this paper we show how the LRT rule can be easily interpreted. Furthermore, we show how the LRT is particularly well suited for the complex trade-off between false positive rate (FP, the probability that the IDS outputs an alarm in case of no actual intrusion), and false negative rate (FN, the probability that the IDS outputs no alarm when an intrusion actually occurs).

Recently, IDS researchers have also proposed and applied other machine learning techniques [31, 13, 18, 46, 28, 26, 27, 40]. Lee *et al.* [31] applied meta-classification to improve accuracy and efficiency, and to make data mining based IDSs more adaptable. Didaci *et al.* [13] and Giacinto and Roli [18] applied three different meta-classification methods proposed in [52] to the outputs of three neural networks trained on different feature sets from the KDD tcpdump data. Kruegel *et al.* [28] proposed to use Bayesian networks to improve the aggregation of different detector outputs. In [26, 27], Kruegel and Vigna proposed several Web anomaly detectors and combined them using weighted voting. In [40], Perdisci *et al.* proposed an ensemble (using majority voting) of several one-class SVM classifiers (using different feature set) to harden payload-based anomaly detection systems. In [6], Bass pointed out that the art and science of (multisensor) data fusion is applicable to intrusion detection, and discussed several challenges in IDS fusion. Shankar [44] applied the data fusion technique to detect and track rapidly propagating intrusions. Valeur *et al.* [47] proposed a comprehensive framework for intrusion detection alert correlation, partially including alert fusion. This work showed that by combining multiple classifiers to detect intrusions, we can improve accuracy to some degree, however, none of these efforts explored the theoretical or practical advantages of the LRT rule.

When combining alerts from different IDSs, we need to distinguish between *alert fusion* [6, 44, 47] and *alert correlation* [47, 12, 41, 39, 11]. The alert fusion problem is the combination of alerts representing independent detection of the same attack occurrence [47], while the alert correlation problem attempts to group alerts and provide a more succinct, high-level view of the intrusions attempts (typically done via attack graphs or attack scenarios). Compared to alert correlation, alert fusion is a very narrow and specific process. However, there is very little work defining a formal and principled framework for alert fusion where the operational performance is evaluated under clearly defined metrics. In this paper we address this concern.

One of our main evaluation metrics is cost. In cost-based analysis there were several efforts. In the machine learning field, Adacost [16] is an enhanced Adaboost that considers cost factors. However the interpretation of the output of an ensemble using these techniques is mostly based only on good empirical results, limiting the understanding that an IDS analyst must have when receiving an alarm. In the intrusion detection area, Lee and Fan [30, 15, 45] were among the first to explore cost-sensitive approaches for intrusion detection. A very detailed case-example of identifying costs in a large network was done by Arora [3]. Although identifying the associated costs is not an easy task, it is essential for risk management; a major component of an overall strategy for information security. In this paper we study the cost-aware structure for the alert fusion problem.

3. DECISION-THEORETIC ALERT FUSION FOR AN IDS ENSEMBLE

3.1 Formal Model of IDS Fusion

In our IDS fusion model, each intrusion detector analyzes the data (e.g., network traffic), and outputs whether it is anomalous or normal. Formally, we can model the observation of each data unit (packet or flow, depending on the analysis unit of the detector) with two status, either normal (H_0) or anomaly (H_1). In the ensemble, we assume each detector have the same unit of analysis (same granularity of alert, see [19] for an extended discussion on handling other cases). Each IDSi ($i = 1, ..., n;\ n \geq 2$) observing the data unit (e.g., network packet) makes its own decision y_i as H_1 (in this case it outputs $y_i = 1$) or H_0 (it outputs $y_i = 0$). All the decisions of individual detectors (which form a decision vector $\vec{Y} = \{y_1, ..., y_n\}$) will be gathered at a fusion center, and a global decision will be made (Figure 1). y_0 is the final decision reported to the network security administrator.

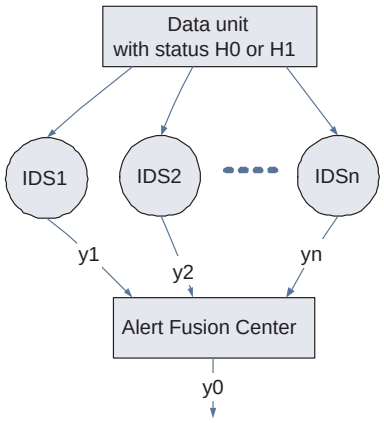


Figure 1: Alert fusion for an IDS ensemble

To perform a cost-based analysis, we need specifically define cost C_{ij}, which is related to a situation in which IDS outputs H_j while H_i is the real output. C_{01} is the cost of FP (i.e., $P(H_1|H_0)$). C_{10} is the cost of FN (i.e., $P(H_0|H_1)$). C_{00} is the cost of true negative ($P(H_0|H_0)$, the probability that the IDS indicates normal when no intrusion occurs), and C_{11} is the cost of detection (also called true positive, $P(H_1|H_1)$, the probability that the IDS outputs an alarm when an intrusion occurs). In most cases, we can just set $C_{00} = C_{11} = 0$ because we assume an IDS involves no cost (in terms of loss of information or services in the protected network) if it does the right thing. (We will use this setting throughout the entire paper.) Now we can define an expected cost function C_{exp}, which reflects the overall Bayesian risk.

$$C_{exp} = \sum_{i=0}^{1}\sum_{j=0}^{1} C_{ij}P_iP(H_j|H_i), \tag{1}$$

where P_i is the prior probability of H_i, $i = 0, 1$. Thus, C_{exp} is the expected mean value of cost over all possible situations. In the IDS literature, P_1 is called base rate [4] that indicates the probability of each data unit being malicious (H_1) (or, the prior fraction of intrusion in the entire data stream). In a cost-based analysis case, C_{ij} should always be pre-defined by administrators according to the risk model of their networks [3]. We will briefly discuss an unbiased estimation method of P_i later in the paper.

We further define the probability of false positive (FP) and the probability of detection ($1 - FN$) corresponding to detector i, denoted as P_{Fi} and P_{Di}, respectively. The overall probability of false positive and the overall probability of detection of the IDS ensemble are denoted as P_F and P_D, respectively.

$$P_F = \sum_{\vec{Y}} P(y_0 = 1|\vec{Y})P(\vec{Y}|H_0)$$

$$P_D = \sum_{\vec{Y}} P(y_0 = 1|\vec{Y})P(\vec{Y}|H_1)$$

Here the sum over $\vec{Y}$ indicates that all the possible values of $\vec{Y}$ be taken and summed up. Then we can expand C_{exp} in Eq(1) using P_F and P_D.

$$\begin{aligned} C_{exp} &= C_{01}P_0P_F + C_{10}P_1(1 - P_D) \\ &= C_{01}P_0P_F - C_{10}P_1P_D + C_{10}P_1 \\ &= C_{01}P_0\textstyle\sum_{\vec{Y}} P(y = 1|\vec{Y})P(\vec{Y}|H_0) \\ &\quad -C_{10}P_1\textstyle\sum_{\vec{Y}} P(y = 1|\vec{Y})P(\vec{Y}|H_1) + C_{10}P_1 \end{aligned} \tag{2}$$

3.2 Decision-Theoretic Alert Fusion Based on LRT

The likelihood ratio test (LRT) is a statistical test of the goodness-of-fit between two test models. Based on LRT technique, which is commonly used in data fusion, we have the following theorem [49, 22]. It provides an optimal decision algorithm for combining alerts of the IDS ensemble.

THEOREM 1. *Given that every IDS makes its own decision, the following fusion rule can minimize the expected cost C_{exp}*

$$\begin{cases} \textit{If} \quad \frac{P(\vec{Y}|H_1)}{P(\vec{Y}|H_0)} > \frac{C_{01}P_0}{C_{10}P_1}, & \textit{Output } y_0 = 1 \\ \textit{If} \quad \frac{P(\vec{Y}|H_1)}{P(\vec{Y}|H_0)} < \frac{C_{01}P_0}{C_{10}P_1}, & \textit{Output } y_0 = 0 \end{cases} \tag{3}$$

This algorithm tells us that if the left part of the equation is greater than the right, then the final decision yields output $y_0 = 1$ (anomaly); otherwise, it yields output $y_0 = 0$ (normal). If they are equal, we can arbitrarily pick either decision.

Note that in order to use this rule, we need to know $\frac{C_{01}P_0}{C_{10}P_1}$ and $\frac{P(\vec{Y}|H_1)}{P(\vec{Y}|H_0}$. We now discuss our interpretation of these parameters.

Finding the exact distribution $P(\vec{Y}|H_i)$ of a multidimensional random variable $\vec{Y}$ is, in general, a very hard problem. In fact, a large body of research in the machine learning community focuses precisely on models to approximate unknown multidimensional distributions [21]. A fundamental result in approximating distributions is the *bias-variance* trade-off. This result says that as the complexity of our model increases (i.e., as the number of parameters increases) the variance of the model also increases, i.e., the approximation to $P(\vec{Y}|H_i)$ is very dependent on the training data, but it cannot generalize to other domains other than the one associated with the collected data. Similarly, if the complexity of the model is small (i.e., if the number of parameters used is very small), the bias of the model will be large, i.e., the model will not have enough parameters to approximate the real distribution. This concept is illustrated in Figure 2.

We therefore make a very common approximation to the joint distribution by assuming *conditional* independence of the detectors:

$$P(\vec{Y}|H_i) = P(y_1|H_i)P(y_2|H_i)\cdots P(y_n|H_i). \tag{4}$$

With this approximation, in order to compute the joint distribution of the ensemble, we only need to estimate $2n$ parameters: n probabilities of detection (P_{Di}), and n probabilities of false alarm (P_{Fi}), as opposed to the $2 \times (2^n - 1)$ parameters required to model the precise multinomial distributions $P(\vec{Y}|H_1)$ and $P(\vec{Y}|H_0)$.

The conditional independence assumption is a very common technique to obtain an approximation of the joint distribution, and this assumption has been applied successfully in several intrusion detection scenarios [3, 28], and in machine learning scenarios (most notably by the naïve Bayes classifier [21]). We will comment more on the practical effects of this assumption in Section 6.

Note that there is a difference between *complete* independence and *conditionally* independence. In our conditional independence assumption, we still allow for correlations such as the fact that if

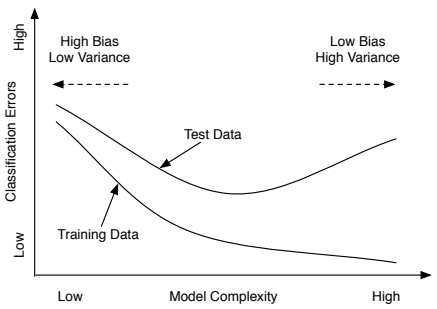


Figure 2: Bias-variance trade-off: with more parameters to model a distribution, we can obtain a more accurate result, but because we have finite training data (and possibly non-stationary distributions), we cannot estimate reliably all these parameters and thus our variance increases.

one detector raises an alarm, this increases the likelihood of the other detector firing an alarm.

Regarding the knowledge of the cost and the base-rate, although several research has presented a very thorough analysis of risks and operating costs [3] (and thus have a reliable estimate for $\frac{C_{01}P_0}{C_{10}P_1}$), such information in general may not be available in practical scenarios. Therefore, we show in Section 5 how to use and interpret the LRT rule even if we do not know these parameters.

For simplicity, it is also convenient to use the log-likelihood ratios. Now we can have the following fusion/decision rule:

$$\begin{cases} If \quad \sum_{i\in S_0} \log \frac{(1-P_{Di})}{(1-P_{Fi})} + \sum_{j\in S_1} \log \frac{P_{Dj}}{P_{Fj}} > \log \frac{C_{01}P_0}{C_{10}P_1}, \ \ y_0 = 1 \\ If \quad \sum_{i\in S_0} \log \frac{(1-P_{Di})}{(1-P_{Fi})} + \sum_{j\in S_1} \log \frac{P_{Dj}}{P_{Fj}} < \log \frac{C_{01}P_0}{C_{10}P_1}, \ \ y_0 = 0 \end{cases} \tag{5}$$

where S_0 is the set of all detectors i, $y_i \in \vec{Y}$, and $y_i = 0$; S_1 is the set of all detectors j, $y_j \in \vec{Y}$, and $y_j = 1$. Note in this form, we can also consider Eq. (5) as a special form of weighted voting.

4. EXPERIMENT EVALUATION

In order to evaluate the performance of our LRT ensemble, we performed several experiments using multiple IDSs on two different data sets to show its practical application. We will provide theoretic reasoning in the next section.

In the evaluation, we will compare the following existing common ensemble decision/fusion rules:

- AND rule: only if all the detectors report anomaly ($y_i = 1$ for all $i = 1..n$), then $y_0 = 1$; otherwise, $y_0 = 0$.
- OR rule: if any detector reports anomaly, then $y_0 = 1$; only if all detectors report normal, then $y_0 = 0$.
- Majority (MAJ) rule: if most (at least half) of the detectors report anomaly, then $y_0 = 1$, otherwise, $y_0 = 0$.
- Weighted voting (VOT) rule: treat the decision $1, 0$ as $1, -1$, and use weighted voting $y_0 = sign(\sum_i \alpha_i y_i')$, here α_i is the weight assigned to detector i and $\alpha_i = \frac{1}{2}\ln(\frac{1-\epsilon_i}{\epsilon_i})$ [17]. This gives more weight on the detector with less classification error rate.
- LRT rule: our cost-aware decision rule based on the LRT fusion algorithm.

These rules are selected because they are simple/intuitive in nature, and most commonly used in machine learning and intrusion detection literature.

4.1 Experiment Using Machine Learning Based IDSs on KDD Data Set

Our first experiment used KDD cup 1999 data set [1], which was produced from the 1998 DARPA Intrusion Detection Evaluation program [33]. The raw data includes about nine weeks of TCP dump network traffic containing normal traffic and many attacks. Lee and Stolfo *et al.* [31] further processed data based on three categories of derived features, i.e., basic features of individual TCP connections, content features within a connection suggested by domain knowledge, and traffic features computed using a two-second time window (please refer to [31] for detail). We acknowledge the limitation/flaw of this data set as criticized in [36, 35]. The reason we choose this as our *initial* study is because it is the only well-studied, documented and public trace for intrusion detection. We will further comment on this shortly after we show the experiment results.

In this experiment, we chose `kddcup.data_10_percent.gz` as the training data set and `corrected.gz` as the testing data set. The training set (75M uncompressed) is about 10% of the full data set and contains 494,020 records. The testing data (45M uncompressed) with corrected labels has 311,029 records, among which 60,593 are normal and 250,436 are intrusions. Every record is labeled only using 0 (normal) or 1 (anomaly) in the experiment.

We used several machine learning based IDSs to construct the IDS ensemble. Specifically, we chose four different machine learning algorithms, i.e., DT (decision tree algorithm, specifically, C4.5), NB (Naïve Bayes classifier), KNN (K-Nearest Neighbor, specifically, we set $k = 9$), and SVM (Support Vector Machine, specifically, we use the LIBSVM [10] tool) [48]. Most of the algorithms are described in [37]. All of them have been successfully applied to intrusion detection [2, 23, 32]. We used all these four machine learning-based IDSs (DT, NB, KNN, and SVM) to build an IDS ensemble.

Table 1 reports the accuracy of each detector and fusion rule. Note that here we have not involved cost factors yet, but simply show accuracy in terms of the detection rate (P_D), the false positive rate (P_F), and the total error rate (ϵ). Since the LRT rule requires cost factors (which define the threshold the LRT rule uses for finding its final decision), it is not shown in the table. From the table we can see that for these four single detectors, DT and KNN achieve slightly better accuracy than NB in terms of P_D, P_F and ϵ. SVM obtains the lowest false positive rate among four, however, also the lowest detection rate.

For the fusion result, the AND rule achieves the lowest false positive rate (0.00066014). The OR rule has the highest detection rate (0.9185) and the lowest total error rate. The MAJ and VOT rules have some balance between false positive rate and false negative rate, compared with the AND and OR rules.

Now we will involve a cost factor analysis. In the experiment, since we have all the running results, we can compute the final total cost related to the total number of false positives and false negatives. We define

$$C_{total} = N_{01}C_{01} + N_{10}C_{10},$$

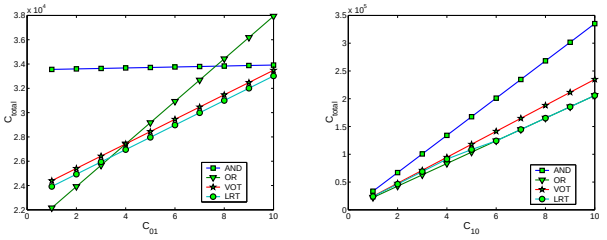
where N_{01} is the number of false positives in testing, and N_{10} is the number of false negatives in testing. Thus, C_{total} reflects the final overall cost.

We want to cover as many risk scenarios as possible (with

Table 1: Accuracy of each detector and fusion rule (ensembling four IDSs)

	DT	NB	KNN	SVM	AND	OR	MAJ	VOT
P_D	0.9103	0.9016	0.9080	0.876	0.8662	0.9185	0.9027	0.9066
P_F	0.017609	0.023204	0.0058423	0.0036473	0.00066014	0.028931	0.0040269	0.016669
ϵ	0.075636	0.083767	0.075212	0.10053	0.10789	0.071254	0.079092	0.078446

different weights of cost for false alarms and missed attacks) that could occur. Assume the minimum unit of cost value is 1, we first fix $C_{10} = 1$ and vary C_{01} to obtain its effect (simulating the cases that false positive is equal or more important than false negative) on different rules. Then, we fix $C_{01} = 1$ and vary C_{10} to simulate the cases that the cost of false negative is equal or greater than that of false positive. Thus, we verify the efforts of these fusion rules in various cost scenarios (with different setting of C_{01} and C_{10}). Figure 3(a) and (b) show C_{total} for different rules in these two settings.



(a) Fix the cost of FN ($C_{10} = 1$) in all the cases, vary the cost of FP (C_{01}).

(b) Fix the cost of FP ($C_{01} = 1$) in all the cases, vary the cost of FN (C_{10}).

Figure 3: Total cost for different rules in different risk scenarios (ensembling four IDSs). The LRT rule obtains the lowest overall cost in most of the scenarios.

Generally speaking, we can still claim that the LRT rule performs the best. The MAJ and VOT rules achieve almost identical performance in the experiment. This is intuitive because all of the these four IDSs have similar total error rates that result in the effect of weights in VOT similar to that in MAJ. Thus, we only draw the VOT rule in the figures. The VOT rule performs well, but still worse than the LRT rule. The AND rule has a nearly stable cost in Figure 3(a). The reason is because it has the lowest false positives so that a slight change of the cost of FP does not have a significant effect. In many scenarios, the OR rule performs as well as the LRT rule (as shown in Figure 3(b)), but it does not perform well when the cost of FP is larger than 4, as indicated in the right part of Figure 3(a). In some cases, the OR rule outperforms LRT (left part of Figure 3(a)). Some possible reasons are as follows:

1. The four IDSs are not ideally independent. If we use Eq(5) and assume independence, we may obtain incorrect joint conditional density of $P(\vec{Y}|H_i)$ when this assumption does not hold.

2. The KDD data set is flawed [36, 35] as we mentioned before. In addition, the distribution of the testing data is greatly different from that of the training data. And the base rate in the testing data set is too high ($P_1 = 0.8$, which is unreal because we know in practice the fraction of intrusion among all data is very low [4]). Since the OR rule can greatly improve the detection rate (reduce false negatives), and since most of the testing data are anomalous, the OR rule can achieve good performance in KDD data set. However, the OR rule also increases the false positives, thus, when the cost of false positive increases, it will cause very high overall cost as indicated in the right part of Figure 3(a).

Due to these reasons, our experiments with the KDD data set could be considered as a *bad (if not the worst)* situation. Yet the LRT technique still achieved good results. Thus, the experiments still confirm that LRT is an effective cost-aware fusion rule (even when the independent assumption may not hold well) and performs the best in most of the risk scenario cases.

Having noticed the flaws of the KDD data set, we conducted another experiment by replacing KDD data set with a real network traffic trace. At the same time, we also used some real IDSs instead of machine learning based IDSs.

4.2 Experiment Using Real IDSs on Real Traces

We collected a real trace at a Web server in our campus network. The trace contains about 30 minutes `HTTP` traffic, around 5 million packets. We divided the trace into two halves, one for training (for anomaly IDSs), and the other for testing. And we injected `HTTP` attacks into the testing set, using tools such as libwhisker [42]. Finally we got a testing set with base rate $P_1 = 0.00082$, which is relatively realistic [4].

We constructed the IDS ensemble using three real IDSs, namely, Snort [43] version 2.3, NetAD (Network Traffic Anomaly Detector, [34]), and PAYL (Payload Anomaly Detection, [50]). Snort is a packet level signature based NIDS. NetAD and PAYL are two packet level anomaly based IDSs. PAYL uses byte frequencies in the payload as the normal profile. NetAD calculates a score based on selected fields in the first 48 bytes (of which 40 bytes are in IP and TCP header, 8 bytes are in the payload) of each packet. If this score exceeds a threshold, then an intrusion alert is issued. Since the original code of NetAD is specific to the DARPA data set, we did a few small modifications and set the rate limiting using 5 second time window instead of 1 minute. Clearly, these three IDSs are very different in their feature sets and detection algorithms. We expect these differences would demonstrate a strong evidence of diversity and independence (but we cannot formally prove it).

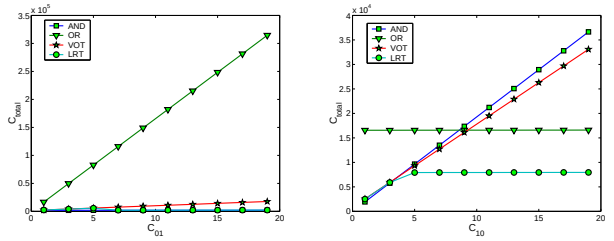
Table 2 shows the accuracy of these three IDSs and some fusion rules (except the LRT rule, which again requires cost factors to define the threshold). We can see that these three IDSs have different performance in terms of their FP and FN. PAYL detected most of the attacks, and Snort produced the least false positives. For the fusion rules, similar to the result from the previous experiment, the AND rule obtains the lowest overall false positive rate but the worst detection rate (zero in this case). The OR rule achieves the highest detection rate as well as the highest false positive rate. The MAJ and VOT rules are still identical in the experiment (although VOT assigns different weight to different IDS, it still generates the identical combinational decision result as MAJ in our data set). Thus, we only show the result of VOT rule.

The overall cost of running different fusion rules on the alerts

Table 2: Accuracy of each detector and fusion rule (ensembling three real IDSs)

	Snort	PAYL	NetAD	AND	OR	MAJ	VOT
P_D	0.016	0.99896	0.1037	0	0.999	0.1198	0.1198
P_F	0.0000237	0.00336	0.004	0.0000025	0.007	0.00035667	0.00035667
ϵ	0.00082758	0.0034	0.0047	0.00082	0.007	0.0011	0.0011

generated by these three real IDSs are shown in Figure 4. We manipulate the cost factors to demonstrate various risk scenarios.



(a) Fix the cost of FN ($C_{10} = 1$) in all the cases, change the cost of FP (C_{01}).

(b) Fix the cost of FP ($C_{01} = 1$) in all the cases, change the cost of FN (C_{10}).

Figure 4: Total cost for different rules in different risk scenarios (ensembling three real IDSs). The LRT rule outperforms other rules (it obtains low overall cost in most of the scenarios.)

From Figure 4(a), we can see that the AND rule achieves similar stable performance as the LRT rule when the cost of FP is greater than the cost of FN. This is because the AND rule can greatly reduce false positives, and thus reducing the total cost (a similar reason discussed above). However, the AND rule also reduces the detection rate; thus, when the cost of false negative increases, it will still cause high total cost (as shown in Figure 4(b)). The OR rule no longer performs well in this experiment. It keeps a relatively stable cost in Figure 4(b) because it has a very low false negative rate; and thus, the slight change of the cost of FN does not significantly affect the total cost. In this experiment, the LRT rule again obtains the lowest total cost in most of the risk situations. We conclude that the LRT rule outperforms other methods in achieving lower cost in our evaluation.

Note that in this experiment, we have a realistically low base rate (i.e., $P_1 = 0.00082$), compared to the higher base rate in the previous experiments using KDD data. Except for the LRT rule, none of other methods can obtain low cost in both experiments. This means the LRT rule is adaptive to the base rate. It is also adaptive to the setting of cost factors. Whatever the cost factors are (whatever the risk scenarios are), it always achieves very low overall cost in our experiments, while other fusion techniques cannot.

5. FORMAL INTERPRETATION AND REASONING ABOUT THE LRT ENSEMBLE

In this section we introduce a formal interpretation to analyze the LRT ensemble rule. To present also the intuition behind the LRT ensemble rule, we make use of *receiver operating characteristic* (ROC) curves. The ROC curve is a two dimensional graph that shows the performance of a detector in terms of its false alarm rate P_F (in the x-axis), and its detection rate P_D (in the y-axis). We say that an ROC curve defined by the probability of detection $f_1(x)$ (where x is a given probability of false alarm) *dominates* an ROC curve defined by $f_2(x)$ if for every x, $f_1(x) \geq f_2(x)$.

The relationship between the ROC curves and the expected cost function defined in Eq. (1) has been studied in the context of intrusion detection in [9]. In particular it is known that the lines with the same expected cost in the ROC space can be represented as lines with slope

$$\tau = \frac{C_{01}P_0}{C_{10}P_1}. \quad (6)$$

Note that τ as defined in Eq. (6) is also the threshold for the rule in Eq. (3). This dual nature of τ will help us understand the intuition behind the LRT ensemble rule.

This relationship between ROC curves and the expected cost implies that the ensemble rule that minimizes the expected cost for any fixed costs (C_{01} and C_{10}) is an ensemble rule that is part of an optimal ROC curve (i.e., an ROC curve that dominates any other ROC curve achievable as a function of ensemble rules). In this section, we show how to derive this optimal ROC curve using the LRT ensemble rule.

5.1 Understanding the LRT Rule

First, we consider the simple case when we have two IDSs available and want to obtain the optimal ensemble from the information available from the two. Assume the performance of IDS_i is (P_{Fi}, P_{Di}), where $i \in \{1, 2\}$. In Figure 5 we can see the ROC curves for two IDSs that we will use in the remaining of this section.

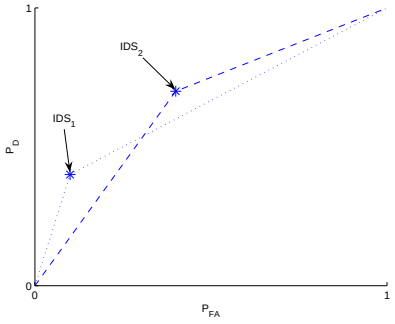


Figure 5: IDS_i is represented as a point (P_{Fi}, P_{Di}). Any point below the dotted lines is suboptimal (i.e., it never minimizes the expected cost function).

Let

$$l(\vec{Y}) = \frac{P(\vec{Y}|H_1)}{P(\vec{Y}|H_0)} \quad (7)$$

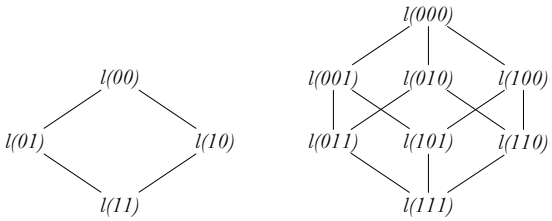
So for example,

$$l(00) = \frac{(1 - P_{D1})(1 - P_{D2})}{(1 - P_{F1})(1 - P_{F2})}$$

and

$$l(10) = \frac{P_{D1}(1 - P_{D2})}{P_{F1}(1 - P_{F2})}$$

Assuming $P_{Fi} < P_{Di}$, it is easy to show that the likelihood function l satisfies the following partial order $l(00) < l(10) < l(11)$ and $l(00) < l(01) < l(11)$. A graphical representation can be seen in Figure 6(a).



(a) Partial order of the likelihood function for the combiation of two IDSs.

(b) Partial order of the likelihood function for the combination of three IDSs.

Figure 6: Partial order of the likelihood function for the combination of IDSs

Given the above partial order for l, there are five regions for τ in $[0, \infty)$ with different optimal rules (examples shown in Figure 7):

- For $0 \leq \tau \leq l(00)$, it is clear from Eq. (3) that the IDS ensemble will always output $y_0 = 1$. Therefore the performance of the ensemble rule is $(P_F, P_D) = (1, 1)$. This operating point can be interpreted as an ensemble that always outputs an alarm, no matter what the decision of each individual IDS is. An example of this operating region can be seen in Figure 7(a).

- For $l(00) \leq \tau \leq \min\{l(01), l(10)\}$,

  $$\Pr\left[l(\vec{Y}) > \tau\right] = \Pr\left[\vec{Y} = 01 \vee \vec{Y} = 10 \vee \vec{Y} = 11\right]$$

  This probability under H_0 equals $P_F = P_{F1} + P_{F2} - P_{F1}P_{F2}$, and under H_1 equals $P_D = P_{D1} + P_{D2} - P_{D1}P_{D2}$. This operating point can be interpreted as the *OR* rule for making an ensemble, since the LRT ensemble outputs an alarm when either IDS_1 or IDS_2 (or both) outputs an alarm. An example of this operating region can be seen in Figure 7(b).

- For $\min\{l(01), l(10)\} \leq \tau \leq \max\{l(01), l(10)\}$, the performance of the ensemble depends on whether $l(01) < l(10)$, which implies $(P_F, P_D) = (P_{F1}, P_{D1})$ or $l(10) < l(01)$, which in turn implies $(P_F, P_D) = (P_{F2}, P_{D2})$[2]. This operating point can be interpreted as choosing one IDS only, i.e., there is always an IDS that performs better than the other, and therefore the LRT ensemble will choose this IDS and ignore the alarms of the other! An example of this operating region can be seen in Figure 7(c).

- For $\max\{l(01), l(10)\} \leq \tau \leq l(11)$,

  $$\Pr\left[l(\vec{Y}) > \tau\right] = \Pr\left[\vec{Y} = 11\right]$$

  This probability under H_0 equals $P_F = P_{F1}P_{F2}$, and under H_1 equals $P_D = P_{D1}P_{D2}$. This operating point for the LRT ensemble can be interpreted as the *AND* rule, since the LRT ensemble will only output an alarm if both, IDS_1 and IDS_2 have output an alarm. An example of this operating region can be seen in Figure 7(d).

- For $l(11) \leq \tau \leq \infty$, the LRT ensemble will never output an alarm. Therefore, $(P_F, P_D) = (0, 0)$. An example of this operating region can be seen in Figure 7(e).

The result of this analysis can be seen in Figure 7(f). Depending on τ, there are five different operating points for an LRT ensemble of two IDS. Output no alarm, Apply the *AND* rule, Choose IDS_1 (in this case), Apply the *OR* rule, and Always output an alarm. We can see in this plot that the operating point of IDS_2 is suboptimal and therefore it cannot minimize the expected cost function. Notice also that for the same false alarm rate as IDS_2, a randomization between the *OR* rule and IDS_1 achieves a better probability of detection.

Note also that since the LRT ensemble depends on τ, it is not a fixed rule but a rule that depends on the costs and the base rate. For example, the *AND* ensemble rule is only optimal for costs and base rates such that $\max\{l(01), l(10)\} \leq \tau \leq l(11)$. Similarly, the *OR* ensemble rule is only optimal for costs and base rates such that $l(00) \leq \tau \leq \min\{l(01), l(10)\}$. Therefore, as we change the costs and the base rate, we expect the optimal operating point of the LRT ensemble to be shifting among several points to minimize the overall cost function, while other ensemble rules can only hope to minimize the cost function for a very small range of operating conditions.

A similar analysis can be done for the combination of more than two IDSs. For example for the ensemble of three IDSs, the *AND* ensemble rule corresponds to $\tau < l(111)$ but greater than the likelihood containing two alerts, and the *OR* ensemble rule corresponds to τ greater than $l(000)$ but smaller than any likelihood containing at least two alerts. With more than two IDSs we can also analyze the the majority rule, which can initially be thought to be the operating point in the LRT ensemble that minimizes the expected cost function for τ greater than any likelihood with a minority of alarms and smaller than any likelihood with a majority of alarms ($\max\{l(001), l(010), l(100)\} < \tau < \min\{l(011), l(101), l(110)\}$).

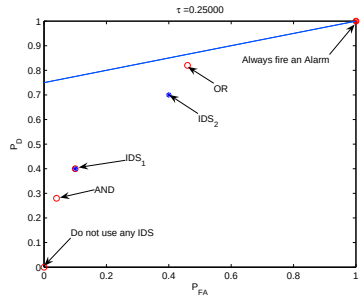
However, the partial order shown in Figure 6(b) shows that in this case the above equation might not always hold. For example, when $l(101) < l(010)$, we should believe more if there is an alarm from IDS_2 alone, than when the other IDSs output an alarm and IDS_2 does not.

Finally, it is important to point out that even in the case when the costs C_{01} and C_{10} are unknown, the LRT ensemble rule can be used by trying different threshold values τ. By selecting different values of τ, the LRT outputs 2^n different ensemble rules. We emphasize that the LRT rule can be used *even if we do not know the costs or the base-rate*; furthermore, even in this case, we still obtain theoretical performance guarantees.

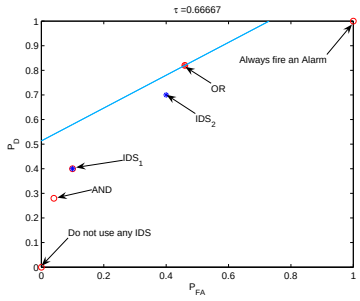
Neyman-Pearson theory [38] states that the LRT for some fixed τ is a more powerful test for its size; therefore, no other test has higher probability of detection for the same bound on the false alarms. The operator of an IDS can then select the appropriate τ based on the performance of the ensemble rules, while having a principled guarantee that by tuning τ, the optimal tradeoff between false alarms and detection rates is achieved.

This 2^n optimal rules considered by the LRT test are a significant reduction from the set of all possible ensemble rules. To quantify

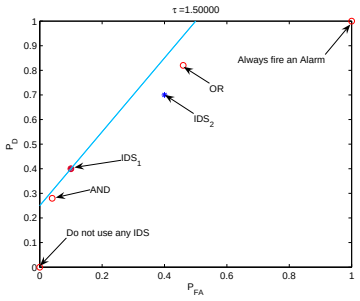
[2]The relation $l(10) = l(01)$ implies that (P_{F1}, P_{D1}) and (P_{F2}, P_{D2}) are in the same performance line of the point for $\tau \in (l(00), \min\{l(01), l(10)\})$ and $\tau \in (\max\{l(01), l(10)\}, l(11))$, and therefore this case can be ignored.



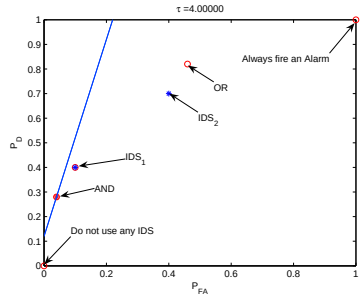
(a) An example for $0 \leq \tau \leq l(00)$



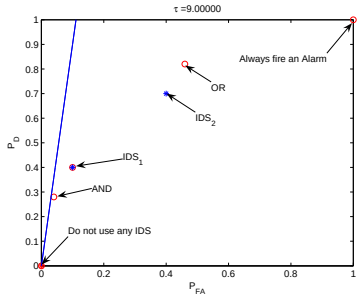
(b) An example for $l(00) \leq \tau \leq \min\{l(01), l(10)\}$



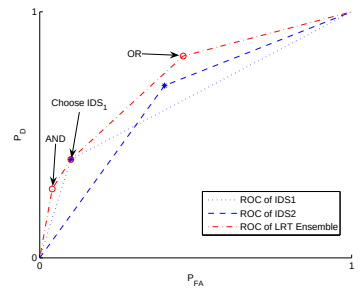
(c) An example for $\min\{l(01), l(10)\} \leq \tau \leq \max\{l(01), l(10)\}$



(d) An example for $\max\{l(01), l(10)\} \leq \tau \leq l(11)$



(e) An example for $l(11) \leq \tau \leq \infty$



(f) The ROC of the LRT ensemble outperforms the ROC of each individual IDS

Figure 7: Examples for different τ

this reduction, note that an ensemble rule is any function that takes inputs from $\vec{Y} \in \{0,1\}^n$ and outputs $y_0 \in \{0,1\}$. Therefore, the size of the set of all possible ensemble rules is 2^{2^n}. Given a particular ordering of the likelihood ratio, the LRT rule excludes $2^{2^n} - 2^n$ suboptimal decision rules. We clarify this fact in the following section.

5.2 Practical Interpretation of the LRT

Note that in practice we do not need to perform the detailed formal analysis of the previous subsection, or compute the partial order. In practice, we only need to compute the likelihood ratios $l(\vec{Y})$ for every possible output $\vec{Y}$ and then sort them.

In order to exemplify this approach, consider again the IDSs in Table 2. In particular, let y_1 denote the output of Snort, y_2 denote the output of PAYL, and y_3 denote the output of NetAD. We can now compute the likelihood ratio for every possible combination of the outputs; for example $l(101) = 182.63$. After computing all likelihood ratios, we obtain the following order:

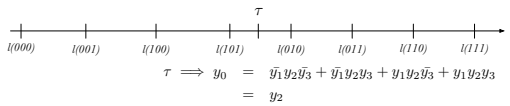
$$l(000) < l(001) < l(100) < l(101) < l(010) < l(011) < l(110) < l(111). \quad (8)$$

The LRT ensemble rule then just tells us the optimal combination of rules that an operator of an IDS can use to obtain optimal performance. Assume for example that the original τ is such that $l(101) < \tau < l(010)$. Then, by looking at Eq. (8), we conclude that the optimal LRT rule is $y_0 = 1$ if and only if

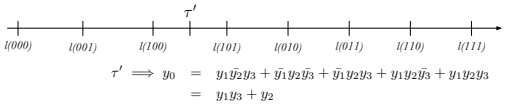
$$\vec{Y} = 010 \vee 011 \vee 110 \vee 111.$$

This LRT rule can be easily simplified using Boolean algebra. Formally, we have: $y_0 = \bar{y_1}y_2\bar{y_3} + \bar{y_1}y_2y_3 + y_1y_2\bar{y_3} + y_1y_2y_3$, which can be simplified to $y_0 = y_2$ (See Figure 8(a)).

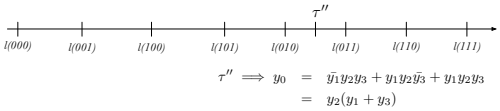
If the original τ misses too many intrusions, then we need to find an ensemble rule that outputs more alarms. A natural choice is to



(a) The optimal rule for $l(101) < \tau < l(010)$ is to output an alarm whenever $y_2 = 1$ (i.e. whenever PAYL outputs an alarm).



(b) The optimal rule for $l(001) < \tau' < l(101)$ is to output an alarm if $y_2 = 1$ or if both $y_1 = 1$ and $y_3 = 1$.



(c) The optimal rule for $l(010) < \tau'' < l(011)$ is to output an alarm if $y_2 = 1$ and this is supported by either $y_1 = 1$ or $y_3 = 1$.

Figure 8: Practical interpretation of the LRT

select a new τ' such that $l(001) < \tau' < l(101)$. By inspecting Eq. (8) again, we conclude that the optimal LRT ensemble rule for τ' is to output an alarm if and only if both y_1 and y_3 output an alarm or if y_2 outputs an alarm. Formally, $y_0 = y_1y_3 + y_2$ (see Figure 8(b)). If we are still missing too many alarms then we can keep decreasing τ.

Similarly, if the original τ generates too many false alarms, then by selecting τ'' such that $l(010) < \tau'' < l(011)$ the optimal rule is to output an alarm whenever $y_2 = 1$ except when both y_1 and y_3 vote against an alarm. Formally, $y_0 = y_2(y_3 + y_1)$ (see Figure 8(c)).

It is also important to note that the ordering given in Eq. (8), excludes rules such as the majority voting rule[3] or the rule $y_0 = y_1$ (to name just a few), because they are *never optimal for any* τ (i.e., they never minimize the expected cost for any C_{01} and C_{10}).

In summary, besides having strong theoretical results, the LRT is also a very practical, intuitive and principled way to combine IDS alerts. The interpretation it provides is that the output of the LRT ensemble rule is an alarm *only when the right combination of IDSs produce alerts*, and where the "right combination" is determined by τ and the ordering of the likelihood ratios.

5.3 A Ranking Interpretation

In the last sections we have shown that even without the knowledge of C_{01}, C_{10} and P_i, the LRT rule gives the most powerful test once we have selected τ based on a desired level of false alarms. In this section we go even further: we show that our method can still be used practically even without the need to set the threshold τ.

Notice that any event in the system can be given a confidence attribute based on its LRT score. With this confidence score the operators of IDSs will examine the most dangerous alerts first, and if they have time, sort through the less suspicious alarms. For example, for the set of IDSs shown in Table 2, the importance of the events can be ordered as in Eq. (8). Thus, all events with $l(111)$ will be the highest-priority events, then all events labeled as $l(110)$, and after this, the events with $l(011)$ etc. Therefore, if an IDS analyst has to decide to investigate a report where only y_1 and y_2 output an alarm, versus an event where only y_2 and y_3 output alarms, the higher score of $l(110)$ over $l(011)$ should indicate that investigating the first report should take precedence over the second. This ranking interpretation will help the IDS analysts to spend their time in the most suspicious events.

6. DISCUSSION

6.1 Estimation of Parameters

FP, FN of IDSs are estimated from trace-driven evaluation, just like any other practically used classifiers. The operator of IDSs may obtain them from the IDS providers, a third party (such as NSS labs, `http://www.nss.co.uk`), or from his own evaluation. In the latter case, the operator tests the IDSs using a sample data from the operation environment (he may also inject attacks). To have a ground truth, the operator could manually examine all the sample data/alerts, or use some sampling strategy (e.g., randomly pick some percentage). In practice, one could avoid manually examining all the sample data by assuming it is mostly benign [50] because the chance of intrusion is probably really low.

Machine learning researchers have given some bounds with certain confidence on the estimation of the true error based on an observed error over a sample of data [37]. Given an observed sample error e_s, with approximately $N\%$ (e.g. 99%) probability, the true error e_t will lie in the very small interval $e_s \pm z_N\sqrt{\frac{e_s(1-e_s)}{n}}$, where n is the number of records in sample data (could be very large to reduce the error bound), z_N is a constant related to the confidence interval $N\%$ we want to reach. For example, if we want a 99% confidence interval then $z_N = 2.58$. Thus, we could assume the estimated FP, FN from the sample data (which is a representative sample of a real situation) are close to the ones in the real situation (in other words, they are relatively stable and can be used in Eq. (5)).

We can use a heuristic approach that estimates the prior distribution of intrusion (the base rate). All we need is an alert rate (r_a) of the IDS, which is the fraction of generated alerts across the entire data set. As we know, this alert rate can be computed as $r_a = P_1(1 - FN) + (1 - P_1)FP$.

Thus, we can approximately estimate the base rate as

$$P_1 = \frac{r_a - FP}{1 - FN - FP}, \tag{9}$$

which provides an unbiased estimation of the real base rate. We then also obtain $P_0 = 1 - P_1$.

The calculation of P_1 can be done periodically during the operation of the IDS if the fusion needs to be online, or done once for the whole history data if the fusion occurs off-line.

6.2 Robustness of LRT Fusion in Tolerating Parameter Estimation Errors

In practice, the estimated false positive rate FP, false negative rate FN may deviate from the exact (real) values (denoted as $\tilde{FP}$,$\tilde{FN}$) to certain degree (e.g., the real data may somewhat vary from the evaluation data, or there is an adaptive adversary). Also, the base rate estimated using Eq. (9) can deviate from the real value due to estimation errors of FP, FN. One may worry about the performance of LRT fusion in these situations, i.e., whether and how robust it is to tolerate possible parameter estimation errors. It is very hard to perform a formal analysis on the effect of error in estimating parameters on the LRT fusion algorithm. Thus, we conducted a set of simulation experiments to see how the LRT can tolerant the error of parameter estimation in IDS alert fusion scenarios.

We now assume the estimated parameters can deviate $\pm\sigma\%$ from the real (exact) values. That is, the estimated FP can vary from $\tilde{FP}(1-\sigma)$ to $\tilde{FP}(1+\sigma)$. Similarly, the estimated FN can lie in $[\tilde{FN}(1-\sigma),\tilde{FN}(1+\sigma)]$. We call σ as the estimation error tolerance bound. We do not assume specific error bound on the base rate, because it is estimated using Eq. (9) so it will definitely involve error due to the errors of FP, FN.

We simulated three distinct detectors (with different detection performance, i.e., different FP, FN) running on a data set with 10,000 events, of which 100 are intrusions (this stands for a base rate of 1%), and generated the alerts. The analysis of the resulted alerts showed that $(\tilde{FP}_1 = 0.052, \tilde{FN}_1 = 0.11)$, $(\tilde{FP}_2 = 0.0182, \tilde{FN}_2 = 0.14)$, $(\tilde{FP}_3 = 0.0089, \tilde{FN}_3 = 0.23)$. We then ran the LRT algorithms on these three alert sets to fuse a final decision. In the process, we assume the FP, FN in use deviate $\pm\sigma\%$ from exact values. We chose different values of σ to test the robustness of LRT on this estimation error bound. For each σ, we ran a thousand times of LRT fusion on the entire alert sets. In each LRT fusion on the entire alert sets, we uniformly and randomly chose FP, FN within the range $FP \in [\tilde{FP}(1-\sigma), \tilde{FP}(1+\sigma)]$, $FN \in [\tilde{FN}(1-\sigma), \tilde{FN}(1+\sigma)]$. The purpose is to see the effect of using (inaccurate) parameters with certain error bound in the LRT

[3]Recall that the majority voting rule is optimal only if there is a τ such that $\max\{l(001), l(010), l(100)\} < \tau < \min\{l(011), l(101), l(110)\}$. However Eq. (8) tells us that for these IDSs, no such τ exists. This makes sense because the values of Table 2 suggest that "outputting an alarm whenever Snort and NetAD vote for an alarm and PAYL votes against it, and not outputting an alarm whenever PAYL votes for it but not the other two IDSs" is not a good rule.

fusion.

We conducted simulation experiments on three cost scenario setting.

- The cost of FN is higher than that of FP (i.e., $C_{10} > C_{01}$). In other words, it is more important to detect all attacks than to obtain fewer false alarms (recall the military network example). We set $C_{01} = 1, C_{10} = 10$ in this case.
- FP is as important as FN, or simply, the cost of FN is equal to that of FP (i.e., $C_{01} = C_{10} = 1$).
- The cost of FN is less than that of FP (i.e., $C_{10} < C_{01}$). In other words, it is more important to reduce false alarms rather than to catch all attacks (recall the example with a single overloaded operator). In this case we set $C_{01} = 10, C_{10} = 1$.

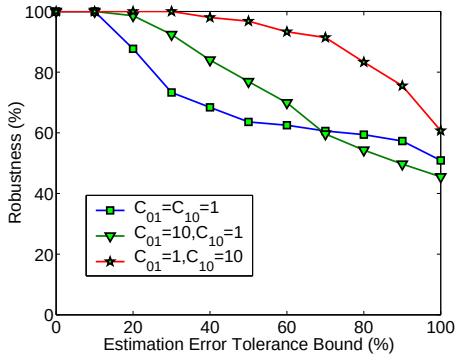


Figure 9: Robustness: simulation result. Estimation errors are allowed. We vary this estimation error tolerance bound. Robustness stands for the probability of having the exact same fusion results on the entire alert sets as using the exact parameters.

The results are shown in Figure 9. We use a robustness metric, which is defined as the probability of having the exact same fusion result on the entire alert data as the situation using the actual $FP/FN/P_1$. The probabilities (parameters) p are chosen uniformly and randomly from $[p*(1-\sigma), p*(1+\sigma)]$ and σ is the estimation error tolerance bound for both FP and FN. For instance, a 100% estimation error ($\sigma = 100\%$) means FN_3 can range from 0 to 0.46, FP_1 can range from 0 to 0.104. Even for the estimation error tolerance bound of 100%, LRT algorithm can achieve the same result on the entire alert data as using the actual parameters with a probability higher than 50% in all the three given cost scenarios. For a reasonably small estimation error tolerance bound (e.g., 20%), LRT algorithm is very robust compared with using the exact parameters.

The simulation results show that the LRT algorithm is a robust fusion technique to tolerate parameter estimation error to a reasonable certain degree.

6.3 The Conditional Independence Assumption

We can relax the conditional independence assumption of Eq. (4) by introducing a more complex approximation to the joint distributions.

A way to introduce more complexity in our approximation is to model the dependence relations between IDSs. We can model the relations using a dependency tree, or more generally, a causal Bayesian network or a graphical model [37, 25, 24]. If the inference relationship can be modeled as a dependance tree, we can perform MIMIC (Mutual-Information-Maximizing Input Clustering) technique [7] to compute the approximate joint distribution. For the construction of causal Bayesian networks, there is work [25, 24] in the machine learning area that can be directly applied to obtain an accurate inference network (dependance relationship), so that we could calculate a more accurate approximation of the joint conditional density.

All these techniques will, however, require substantial efforts by any IDS operator implementing a fusion rule. These additional efforts might deter the IDS operators from implementing more complex approximations to model the joint distribution.

As it stands, the conditional independence assumption is the most practical way to approximate the joint conditional density since any IDS operator can use this approximation by only estimating the performance of each individual IDS. While the conditional independence assumption might not guarantee theoretically optimal rules, it is still an efficient and principled method for fusing information, as shown in our experiments and in other practical applications [3, 28, 37, 21].

Creating more accurate models that avoid the "high variance" problem, and comparing their effectiveness as fusion rules in IDSs against more practical approximations (such as the conditional independence assumption), is a challenging problem and one we plan to explore in future work.

6.4 A Robust Consideration for Cost Evaluation

We showed in previous section that the LRT technique is very robust to parameter estimation errors up to a certain degree. Here we propose another robust way to evaluate the expected (or total) cost to address the concern of a possible large deviation of estimation errors and uncertainties of the parameters in practice. For a robust evaluation with uncertain parameters, we again consider the situation where the estimated base rate, false positive rate and false negative rate deviate from the actual values to certain degree. Now instead of calculating the expected (or total) cost with a static setting of parameters, we use a range (bounded with largest possible deviation to tolerate estimation errors), and among all possible final cost values, we take the highest value (which stands for the worst cases with all possible situation of estimated parameters) as the final result. By doing this, we are actually finding the highest cost against the worst situation with worst possible estimation errors (similar to the idea in [9]). Thus, we can make sure that this final cost is robust in a sense that it is the upper-bound in all cases of possible estimated range of parameters. The real cost evaluation is guaranteed to be better than this robust result given the largest possible estimation error bound.

6.5 Runtime Performance

The runtime performance of our LRT fusion rule is efficient. For example, we test the running time in the first experiment using four IDSs. It only takes 0.187 seconds to finish LRT fusion processing on all 311,029 records, which is about 0.6 *microseconds* per record (on a machine with 512M memory, 2.4GHz Pentium IV processor, Windows 2K OS). Although it is poorer than the performance of the AND rule and the OR rule (about 0.2 microsecond/record), the runtime of our LRT fusion is almost as good as that of the VOT fusion rule, which is about 0.5 microsecond per record. We can improve the computation efficiency if we calculate the log probabilities and thresholds once off-line, store them and simply

use them in the actual fusion decision (instead of calculating again). In addition, if we consider the fact that the fusion is primarily performed *only* when there are alerts generated by IDSs (or even performed off-line), the LRT rule is computationally efficient enough for practical usage.

6.6 Multi-class Extension

The LRT technique discussed so far is on two-class (anomaly or normal) alert fusion. It can also be extended to multi-class situation. The extension is similar to the method of extending two-class SVM to multi-class cases, i.e., we can use a tree structure to perform LRT along some tree path (i.e., it is a sequence of two-class LRT runs). For example, we run the first LRT to decide whether it is class 1 or not, the second LRT to decide whether it is class 2 or not, and so on. The running time (the number of LRT run) is at most $O(m)$, at least $O(log(m))$, where m is the number of classes to classify.

7. CONCLUSION AND FUTURE WORK

In this paper, we proposed a decision-theoretic alert fusion technique for an IDS ensemble and reported our empirical experience from using this technique in practice. We provided formal interpretation of the LRT ensemble based on ROC curve analysis, and discussed the reasoning of why it is better than other approaches. We empirically verified its effectiveness in practice through experiments using several machine learning based IDSs and real IDSs on multiple data sets. The theoretical reasoning and empirical results show that this approach outperforms other existing fusion techniques in practice in terms of achieving low overall cost. Furthermore, this technique is adaptive to different base rates and different risk scenarios (or cost models). In addition, in our simulation test, we showed that this approach can tolerate a reasonable bound of parameter estimation error.

It is also important to notice that the LRT fusion rule is not only an optimal rule in the sense that it minimizes the expected cost, but also optimal in a Neyman-Pearson [38] way. That is, the LRT rule maximizes the probability of detection for a given upper bound on the false alarm rate. The intuition for this notion of optimality can be reflected in the analysis done in Section 5, where the ROC of the ensemble was shown to have superior performance to the ROC of the individual detectors. However, due to space constraints we omit the detail analysis of this property in this paper. For future work, we will study robust and realistic approaches to involve probabilistic techniques and inference models. And we plan to further explore how to obtain better independent (diverse) detectors, which is a very important and interesting problem for IDS research.

The LRT ensemble rule is not limited to applications in intrusion detection. However, we have not encountered a formal treatment of the rule as presented in this paper in other applications such as machine learning. We are currently exploring its use in more general classification systems and voting algorithms, and plan to extend this work to other applications.

8. ACKNOWLEDGMENTS

This work is supported in part by NSF grant CCR-0133629, the Army Research Office contract W911NF0510139 and by TRUST, which receives funding from NSF grant CCF-0424422. The contents of this work are solely the responsibility of the authors and do not necessarily represent the official views of NSF and the U.S. Army.

9. REFERENCES

[1] Kdd cup 1999 data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 2005.

[2] Nahla Ben Amor, Salem Benferhat, and Zied Elouedi. Naive bayes vs decision trees in intrusion detection systems. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 420–424, New York, NY, USA, 2004. ACM Press.

[3] Anish Arora, Dennis Hall, C. Ariel Pinto, Dwayne Ramsey, and Rahul Telang. Measuring the risk-based value of it security solutions. *IT Professional*, 6(6):35–42, Nov.-Dec. 2004.

[4] S. Axelsson. The base-rate fallacy and its implications for the difficulty of intrusion detection. In *Proceedings of ACM CCS'1999*, November 1999.

[5] Marco Barreno, Alvaro A. Cardenas, and J. D. Tygar. Optimal roc curve for a combination of classifiers. In *Proceedings of Neural Information Processing Systems (NIPS) 20*, 2008.

[6] Tim Bass. Intrusion detection systems and multisensor data fusion. *Commun. ACM*, 43(4):99–105, 2000.

[7] J. De Bonet, C. Isbell, and P. Viola. Mimic: Finding optima by estimating probability densities. *Advances in Neural Information Processing Systems*, 9, 1997.

[8] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[9] Alvaro Cardenas, John Baras, and Karl Seamon. A Framework for the Evaluation of Intrusion Detection Systems. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, Oakland, California, May 2006.

[10] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[11] F. Cuppens and A. Miege. Alert correlation in a cooperative intrusion detection framework. In *Proceedings of IEEE Symposium on Security and Privacy 2002*, 2002.

[12] Herve Debar and Andreas Wespi. Aggregration and correlation of intrusion-detection alerts. In *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID'01)*, 2001.

[13] Luca Didaci, Giorgio Giacinto, and Fabio Roli. Ensemble learning for intrusion detection in computer networks. http://citeseer.ist.psu.edu/533620.html.

[14] Thomas G. Dietterich. Ensemble methods in machine learning. *Lecture Notes in Computer Science*, 1857:1–15, 2000.

[15] W. Fan, W. Lee, S. Stolfo, and M. Miller. A multiple model cost-sensitive approach for intrusion detection. In *Proceedings of The Eleventh European Conference on Machine Learning (ECML'00)*, 2000.

[16] W. Fan, S. Stolfo, and J. Zhang. Adacost: cost-sensitive boosting. In *Proceedings of International Coference on Machine Learning (ICML'99)*, 1999.

[17] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Thirteenth International Conference on Machine Learning (ICML)*, pages 148–156, 1996.

[18] G. Giacinto and F. Roli. Intrusion detection in computer networks by multiple classifier systems. In *Proceedings of 16th International Conference on Pattern Recognition (ICPR 2002)*, 2002.

[19] Guofei Gu, Prahlad Fogla, David Dagon, Wenke Lee, and

Boris Skoric. Measuring intrusion detection capability: An information-theoretic approach. In *Proceedings of the 2006 ACM Symposium on Information, Computer, and Communication Security (ASIACCS'06)*, March 2006.

[20] Guofei Gu, Prahlad Fogla, David Dagon, Wenke Lee, and Boris Skoric. Towards an information-theoretic framework for analyzing intrusion detection systems. In *Proceedings of the 11th European Symposium on Research in Computer Security (ESORICS'06)*, September 2006.

[21] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer-Verlag New York, Inc., 2003.

[22] Imad Y. Hoballah and Pramod K. Varshney. Distributed Bayesian signal detection. *IEEE Transactions on Information Theory*, 35(5):995–1000, 1989.

[23] Wenjie Hu, Yihua Liao, and V. Rao Vemuri. Robust support vector machines for anomaly detection in computer security. In *Proc. 2003 International Conference on Machine Learning and Applications (ICMLA'03)*, 2003.

[24] Finn V. Jensen. *Bayesian Networks and Decision Graphs*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2001.

[25] Michael I. Jordan, editor. *Learning in graphical models*. MIT Press, Cambridge, MA, USA, 1999.

[26] C. Kruegel and G. Vigna. Anomaly Detection of Web-based Attacks. In *Proceedings of the* 10^{th} *ACM Conference on Computer and Communication Security (CCS '03)*, pages 251–261, Washington, DC, October 2003. ACM Press.

[27] C. Kruegel, G. Vigna, and W. Robertson. A Multi-model Approach to the Detection of Web-based Attacks. *Computer Networks*, 48(5):717–738, August 2005.

[28] Christopher Kruegel, Darren Mutz, William Robertson, and Fredrik Valeur. Bayesian Event Classification for Intrusion Detection . In *Proceedings of the Annual Computer Security Applications Conference (ACSAC 2003)*, Las Vegas, NV, December 2003.

[29] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley, 2004.

[30] W. Lee, W. Fan, M. Miller, S. Stolfo, and E. Zadok. Cost-sensitive modeling for intrusion detection and response. *Journal of Computer Security*, 10(1,2), 2002.

[31] Wenke Lee and Salvatore J. Stolfo. A framework for constructing features and models for intrusion detection systems. *ACM Transactions on Information and System Security (TISSEC)*, 3(4):p.227–261, 2000.

[32] Yihua Liao and V. Rao Vemuri. Using text categorization techniques for intrusion detection. In *11th USENIX Security Symposium, August 5–9, 2002.*, pages 51–59, 2002.

[33] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. P. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, , and M. A. Zissman. Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation. In *Proceedings of the 2000 DARPA Information Survivability Conference and Exposition (DISCEX'00)*, 2000.

[34] M. Mahoney. Network traffic anomaly detection based on packet bytes. In *Proceedings of 18th ACM Symp. on Applied Computing*, pages 346–350, November 2003.

[35] M. Mahoney and P. Chan. An analysis of the 1999 darpa/lincoln laboratory evaluation data for network anomaly detection. In *Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection (RAID'03)*, 2003.

[36] John McHugh. Testing intrusion detection systems: A critique of the 1998 and 1999 darpa off-line intrusion detection system evaluation as performed by lincoln laboratory. *ACM Transactions on Information and System Security*, 3(4), November 2000.

[37] Tom Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[38] J. Neyman and E. S. Pearson. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London, Series A, Containing Papers of a Mathematical or Physical Character*, 231:289–337, 1933.

[39] Peng Ning, Yun Cui, and Douglas S. Reeves. Constructing attack scenarios through correlation of intrusion alerts. In *Proceedings of the 9th ACM Conference on Computer & Communications Security (CCS'02)*, 2002.

[40] Roberto Perdisci, Guofei Gu, and Wenke Lee. Using an ensemble of one-class svm classifiers to harden payload-based anomaly detection systems. In *Proceedings of the IEEE International Conference on Data Mining (ICDM'06)*, December 2006.

[41] Phillip A. Porras, Martin W. Fong, and Alfonso Valdes. A mission-impact-based approach to infosec alarm correlation. In *Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID'02)*, 2002.

[42] Rain Forest Puppy. Libwhisker official release v2.1, 2004. Available at http://www.wiretrip.net/rfp/lw.asp.

[43] Martin Roesch. Snort: Lightweight intrusion detection for networks. In *LISA*, pages 229–238, 1999.

[44] M. Shankar, N. Rao, and S. Batsell. Fusing intrusion data for detection and containment. In *Proceedings of MILCOM2003*, 2003.

[45] Sal Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Phil Chan. Cost-based modeling for fraud and intrusion detection: Results from the jam project. In *Proceedings of the 2000 DARPA Information Survivability Conference and Exposition (DISCEX '00)*, 2000.

[46] Eric Totel, Frederic Majorczyk, and Ludovic Me. COTS diversity intrusion detection and application to web servers. In *Proceedings of RAID'2005*, September 2005.

[47] F. Valeur, G. Vigna, C.Kruegel, and R. Kemmerer. A Comprehensive Approach to Intrusion Detection Alert Correlation. *IEEE Transactions on Dependable and Secure Computing*, 1(3):146–169, July-September 2004.

[48] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[49] P. Varshney. *Distributed Detection and Data Fusion*. Spinger-Verlag, New York, NY, 1996.

[50] Ke Wang and Salvatore J. Stolfo. Anomalous payload-based network intrusion detection. In *Proceedings of RAID'2004*, September 2004.

[51] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

[52] L. Xu, A. Krzyzak, and CY Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. Systems Man Cybernet*, 22(3):418–435, 1992.