Intrusion Detection based on Distance Combination

Joffroy Beauquier, and Yongjie Hu

***Abstract*—**The intrusion detection problem has been frequently studied, but intrusion detection methods are often based on a single point of view, which always limits the results. In this paper, we introduce a new intrusion detection model based on the combination of different current methods. First we use a notion of distance to unify the different methods. Second we combine these methods using the Pearson correlation coefficients, which measure the relationship between two methods, and we obtain a combined distance. If the combined distance is greater than a predetermined threshold, an intrusion is detected. We have implemented and tested the combination model with two different public data sets: the data set of masquerade detection collected by Schonlau & al., and the data set of program behaviors from the University of New Mexico. The results of the experiments prove that the combination model has better performances.

***Keywords*—**Intrusion detection, combination, distance, Pearson correlation coefficients.

I. Introduction

INTRUSION detection systems (IDS) are either based on misuse detection, trying to detect in the observed behaviors a set of signatures gathered from previous attacks [6], [11], or on anomaly detection, trying to detect any deviation from a "normal behavior" [2], [8], [10]. The drawback of the first one is that it cannot detect a new attack or an attack whose signature has been slightly modified. For circumventing this problem, anomaly-based methods use another approach. For each user, a set of features is extracted from its audit trail (the profile) and a tested behavior is compared to these features. Then unexpected attacks can be detected. In this paper we are interested in anomaly-based methods.

As a matter of fact, there are already many anomaly-based methods based on different profiles. Some of them focus on detecting the "masqueraders", defined as security attacks in which an intruder mimics a legitimate user to access or damage objects. A typical example of a masquerade is a hacker who has gained a legitimate user's password. The problem of masquerade detection has been extensively studied. Schonlau & al [12] have summarized six approaches: Uniqueness, Bayes one-step Markov, Hybrid multi step Markov, Compression model, Incremental probabilistic action modeling (IPAM), and Sequence-match. Maxion [9] uses Naive Bayes classification. Coull [1] presents a method based on comparison of sequences.

In addition, some researchers addressed the problem of detecting the intrusions by profiling the program behaviors. Forrest & al [3] introduced an intrusion detection method for program behaviors (the traces of system calls used by active, privileged processes). Since then, a lot of techniques have been developed focusing on the intrusion detection concerning programs. These approaches can be divided into four categories [15]: n-grams method [3], [14], frequency-based methods, finite state machines [15] and data mining approaches [7].

Looking at the results of experiments for the current methods, it is difficult to say that a method is better than another, because each method has its own advantages, and their results strongly depend on the set of observed behaviors. Moreover each method is based on a single point of view. For instance, uniqueness is based on the frequency of a command [12], Bayes one-step Markov uses the one step transitions from a command to another [12], finite state machines pay attention to the transitions of an individual state in function of some number of previous states [4], [14], etc. In this paper we attempt to combine these different methods.

In a perspective of combination, two problems have to be solved. First, unify the different methods, which are based on different ideas and use different techniques. Second, extract the advantages of each combined method efficiently. The approach that we present is based on a notion of "distance". The distance can be considered as an indicator of the dissimilarity between two observed behaviors. A behavior that is very near to the normal behaviors is considered as normal, while a behavior that is far from the normal behaviors is more likely to be an intrusion. If the users had always the same behaviors, things would be simple: any deviation from the correct behaviors would mean an intrusion. Because the user behaviors are generally changing, some tolerance has to be included in the detection mechanism. In this paper, the bound between acceptance and refusal is determined by a predefined threshold. Any behavior whose distance to the normal behaviors exceeds the threshold is considered as intrusive and the system activates an alarm, while the behaviors whose distance to the normal behaviors is smaller than the threshold are considered as normal behaviors (see Fig. 1).

The first contribution of the paper is the definition of a combined distance. As a matter of fact, several methods already use the notion of distances [14], but it is not always the case. But even in the cases where the distance is not explicit,

Manuscript received August 31, 2007.

J. Beauquier is with LRI, Paris Sud University & INRIA CO 91405 Orsay France (e-mail: jb@lri.fr).

Y. Hu is with LRI, Paris Sud University & INRIA CO 91405 Orsay France (e-mail: hu@ lri.fr).

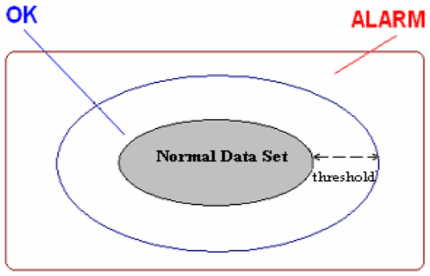


Fig. 1 intrusion detection model based on distance and threshold

we show that it is always possible to find an underlying distance, proving the generality of the approach.

The second contribution of the paper is a method for combining different methods. The combination that we propose is based on a measure of “correlation” between the different methods. The different distances associated to different methods are related according to their correlation and provide a unique combined distance.

The quality of an IDS can be measured by the probabilities of false and missing alarms. A false alarm happens when the IDS decides that the activity of a legitimate user is an intrusion and then generates an alarm, while a missing alarm occurs when an attack is not detected. We use these criteria to compare the results of original experiments with those of our combination (with computed threshold) on the same data sets. Concerning the missing alarm rate the results show a drastic improvement. The results concerning false alarm rate are not as good as for the best methods, but overall stay in an acceptable average. It means that the combination method (on the data we used) detects much more intruders, and produces just a little more false alarms.

The plan of the paper is the following. We first present several related works, then we describe the basic methods that we use in combination and define their associated distances. Then we explain our combination approach, based on techniques like Pearson correlation coefficients. At the end we present the experiments and discuss the results.

II. Related Works

We note that the idea of combining a collection of methods is not new, as several approach were proposed in the past several years. Many of the previous works use majority voting or weighted-voting approaches to integrate the different methods [13], [17], [18]. The idea of majority voting and weighted-voting is to combine the decisions by predicting the result with the highest vote. Our approach, based on Pearson correlation of the predefined distances of the methods, is entirely different. Experiments show that our approach gives much better results (see section 5).

Christopher & al [2] use several different models (i.e. the string length of an attribute value, the presence and absence of a particular attribute, etc.) to detect attacks against Web servers and Web-based applications. The SRI IDES (International’s real time intrusion detection expert system) [5] profiles the normal behaviors of the normal users with different criteria (i.e. CPU time, file accesses or terminals used to log on), and then synthesize the different aspects using their correlation. They are based on a set of fixed, predetermined criteria, while our method can use any possible criterion.

Fan & al. proposed an adaptive combination approach [16]. The core idea of the combination is that one method is used to detect the known attacks, and another method tries to detect the new attacks. However, this approach is based on the assumption that the first method can detect the known attacks accurately. It is obvious that this assumption is dubious and non realistic since the intrusions are always multivariate in the real world.

In summary, there are two main differences between the previous works and ours. First other approaches do not express the notion of distance explicitly. Our approach consists in combining the distance approaches to obtain the definition of a new distance that depends on the previous one, and the resulting distance is defined by the Pearson correlation coefficients. Second, these approaches use fixed criteria and fixed methods for evaluating these criteria for a normal behavior. In our approach, this aspect is completely parameterized. We can use any intrusion detection method in the literature and combine it with some others. The advantage is that our approach always uses the best-known methods at a given time.

III. Anomaly-Based Intrusion detection Methods

Before presenting the basic methods that we combine and their associated distances, we recall the concept of distance.

A. Distance

A distance on a given set G is an application d: $G \times G \rightarrow R$, where *R* denotes the set of real numbers, such that for x, y, z є G:

· $\forall x, y \in G$, d(x, y) ≥ 0 (Non-negative) ;

· $\forall x, y \in G$, d(x, y)=0 ⇔ x=y (Identity) ;

· $\forall x, y, z \in G$, d(x, y) + d(y, z) ≥d(x, z) (Triangle inequality) .

The notion of distance is a common tool for measuring dissimilarity. Many distance functions have been used in different contexts. Here are those we use in the paper:

For two vectors *X= (x₁ ,…, xₙ)* and *Y= (y₁ ,…, yₙ)*:

- Manhattan distance: $d_{Mah}(X,Y) = \sum_{i=1}^{n} |x_i - y_i|$
- Max distance: $d_{\max}(X,Y) = \max(|x_1 - y_1|, |x_2 - y_2|, ..., |x_n - y_n|)$
- Chi-Square distance: $d_{ChiS}(X,Y) = \sqrt{\sum_{i=1}^{n} \frac{|x_i - y_i|^2}{x_i + y_i}}$
- Kullback-Leibler divergence (KLD) distance: $d_{KLD}(X,Y) = \sum_{i=1}^{n} x_i \log \frac{x_i}{y_i}$
- Hamming distance: the number of different positions in two sequences (for instance d_{Ham} *(abca, abaa)*= 1).

We recall that in an IDS, the anomaly-based methods generally use two data sets: the training data set and the test data set. The training data (noted TR) are the observed correct behaviors used to build the profiles, and the test data (noted TE) are the unknown behaviors which must be analyzed. For any observed behavior b in TE, the distance function *d(b,TR)* is used to determined the dissimilarity between b and TR. The way the distance function is defined affects the false and missing alarm rates directly. It is expected for the distance function to return a small value for a non-intrusive behavior and a large value elsewhere.

Now we present the methods based on different criteria that we combine.

B. Overview of Methods

1. Uniqueness

The uniqueness method is based on the notion of uncommon and uniquely used commands. The commands which are not previously seen in the training data reveal an attempted attack [12]. Among the methods that we combine, this method is the best performer in terms of false alarms, but fails for missing alarms.

Let U_{Ck} be the number of users having used command C_k in the training data TR, U the total number of users, K the total number of distinct commands in TR and in an observed behavior b. Let N_{uCk} be the number of times user u uses command C_k in TR and let n_{bCk} be the corresponding value of b. Let $N(u)$ be the total number of commands in user u's training data and $n(b)$ the corresponding value of b. For a command C_k, the "uniqueness score" for b is defined by:

$$uniqueness(C_k) = (1 - \frac{U_{C_k}}{U})(1 - \frac{N_{uC_k}/N(u)}{\Sigma_u N_{uC_k}/N(u)})\frac{n_{bC_k}}{n(b)}$$

To show the accumulative anomalous degree, the distance between the training data TR and b is defined using the Manhattan distance.

$$d(b,TR) = \sum_{k=1}^{K}(1 - \frac{U_{C_k}}{U})(1 - \frac{N_{uC_k}/N(u)}{\Sigma_u N_{uC_k}/N(u)})\left|\frac{N_{uC_k}}{N(u)} - \frac{n_{bC_k}}{n(b)}\right|$$

2. Bayes One Step Markov

The Bayes One-Step Markov approach is based on the transition probabilities from a command to the next. An alarm is activated when the probability of the current transition is not consistent with the transition probabilities generated by the training data [12]. This model has a good performance in terms of missing alarms, but it is not as good for false alarms.

Two hypotheses are made:

For any observed behavior $b=C_1C_2\ldots C_V$,

- H_0: the transition probabilities in b are the same as the corresponding probabilities in the training data;
- H_1: there exists a Dirichlet distribution parameterized by a vector a_0 of nonnegative real number, such that for any b, b respects the Dirichlet distribution.

Let $P(C_1C_2 \ldots C_V|H_0)$ be the conditional probability of b with respect to H_0, and let $P(C_1C_2 \ldots C_V|H_1)$ be the conditional probability of b with respect to H_1. The Bayes factor (BF) of b is the ratio:

$$BF_b = P(C_1C_2 ...C_V \mid H_1)/P(C_1C_2 ...C_V \mid H_0)$$

BF is an indicator of the validity of H_0 with respect to H_1.

Let p_{ujk} be the transition probability from C_j to C_k in TR, let N_{ujk} be the number of times that the pair $(C_j\, C_k)$ appears in user u's TR, let n_{bjk} be the corresponding value in b, let $N(u)$ be the total number of commands in u's TR, let $n(b)$ be the total number of commands in b, let N_{uCk} be the number of times u uses C_k in TR, let n_{bCk} be the number of times of C_k in b and let a_{0k} and a_0 be the Dirichlet distribution parameters, where $a_{0.}=\sum a_{0k}$. It turns out that the Bayes factor of b for $(C_j\, C_k)$ can be computed as [12]:

$$BF_b(C_jC_k) = a_{0k}(a_{0k}+1)...(a_{0k}+n_{bC_k}-1)/a_{0.}(a_{0.}+1)...(a_{0.}+n_b-1)\,p_{ujk}{}^{n_{bjk}}$$

The Bayes factor of TR for $(C_j\, C_k)$ is:

$$BF_{TR}(C_jC_k) = a_{0k}(a_{0k}+1)...(a_{0k}+N_{uC_k}-1)/a_{0.}(a_{0.}+1)...(a_{0.}+N_u-1)\,p_{ujk}{}^{N_{ujk}}$$

The distance between b and TR is defined using the Chi-Square distance:

$$d(b,TR) = \sqrt{\sum_{j,k}\frac{\left(\log BF_{TR}(C_jC_k) - \log BF_b(C_jC_k)\right)^2}{|\log BF_{TR}(C_jC_k) + \log BF_b(C_jC_k)|}}$$

3. Naive Bayes Method

The Naive Bayes method assumes that the commands in a given behavior set are chosen independently. For the k^{th} command c_k of an observed behavior b, let n_{bCk} be the number of times that c_k appears in b and let $P_u(c_k)$ be the probability of c_k in u's *TR*. The conditional probability of b with respect to user u is:

$$p(b \mid u) = \prod_k p_u(c_k)^{n_{bc_k}}$$

Let N_{uCk} be the number of times that u uses the command C_k in TR. For the given user u, the probability of TR is:

$$p(TR \mid u) = \prod_k p_u(C_k)^{N_{uC_k}}$$

Taking the logarithm:

$$\log p(b \mid u) = \sum_k n_{bC_k} \log p_u(c_k)$$

$$\log p(TR \mid u) = \sum_k N_{uC_k} \log p_u(C_k)$$

The Naive Bayes model can use a variant probability to compute $p_u(C_k)$ for ensuring that there are no zero counts [15]. Let A_u be the number of different commands in u's TR, and let α be an arbitrary positive real number (in this model 0.01). By definition, the variant probability $p_u^{NB}(C_k)$of C_k for u is:

$$p_u^{NB}(C_k) = \frac{N_{uC_k} + \alpha}{N(u) + (\alpha \times A_u)}$$

k being the total number of distinct commands in TR and b, the distance between b and TR is defined using the Chi-Square distance:

$$d(b,TR) = \sqrt{\sum_{k=1}^{K}\frac{(N_{uc_k} - n_{bc_k})^2}{N_{uc_k} + n_{bc_k}}\left|\log\left(p_u^{NB}(C_k)\right)\right|}$$

4. Probabilistic Finite State Automata Method

A probabilistic finite state automaton (PFSA) is a finite automaton in which each transition has been equipped with a probability (the sum of all outgoing probabilities is 1 for each state). The probability associated to each transition corresponds to the probability to reach the state from a neighboring state using the transition. In [4] Freeman uses the notion of user signature represented as a PFSA. Each transition in the PFSA corresponds to a user command. The method can be used both for the detection of user behaviors and program behaviors.

For any given finite behavior set H, a PFSA is built in the following way:

First an initial state is chosen. From the initial state a transition is added for any initial commands in H and as many new states are created. The transition for a command C_k receives as a probability the frequency of command C_k among the initial commands.

Then the process is iterated for each new state. This construction can be seen as building the labeled tree classically associated to H (considered as a formal language) and then associating to each edge the frequency of the label among the outgoing labels.

The probability of each sequence in a PFSA is the product of the probabilities of all of its commands.

For instance consider the set of behaviors: H ={login*open*read, login*open*read, login*open*read, login*cd*vi, ls*mail*exit, ls*mail*exit, ls*mail*exit, ls*mail*exit, ls*picp*mv, ls*picp*mv}. The PFSA is represented in Fig. 2.

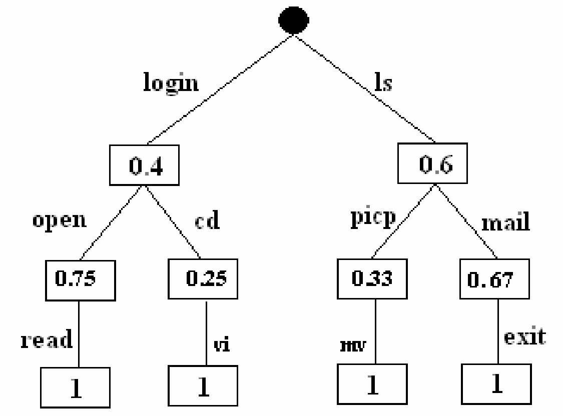


Fig. 2 An example of probabilistic finite state automaton

We denote $PFSA_{TR}$ and $PFSA_{TE}$ the PFSA built from the training data and from the test data respectively. We explain now how to compute the probability of a sequence $s=c_1c_2\ldots c_V$ in the training data, that is not in $PFSA_{TR}$, as follows:

(1) Determine the longest prefix of s in $PFSA_{TR}$. Let $c_1c_2\ldots c_{i-1}$ be the longest prefix.

(2) Check whether or not c_i appears in $PFSA_{TR}$. If c_i is in $PFSA_{TR}$, its probability is a fraction of the frequency of c_i in $PFSA_{TR}$ (in our experiments we choose the ratio 0.01). If not, the probability of ci is a constant (here 0.00001).

(3) Repeat 2 from c_{i+1} to c_V until the end of the sequence s.

The distance between $PFSA_{TR}$ and a sequence in $PFSA_{TE}$ is defined using the Symmetric Kullback-Leibler Divergence (KLD). Let $P_{TR}(s)$ be the probability of sequence s in $PFSA_{TR}$ and let $P_{TE}(s)$ be the probability of s in $PFSA_{TE}$.

$$d_{KLD}(s,TR)=\frac{1}{2}(P_{TR}(s)\log\frac{P_{TR}(s)}{P_{TE}(s)}+P_{TE}(s)\log\frac{P_{TE}(s)}{P_{TR}(s)})$$

The distance between $PFSA_{TR}$ and $PFSA_{TE}$ is defined using either the Symmetric Kullback-Leibler Divergence (KLD) or the max distance.

$$d_{KLD}(PFSA_{TR},PFSA_{TE})=\frac{1}{2}\sum\nolimits_{S}(P_{TR}(s)\log\frac{P_{TR}(s)}{P_{TE}(s)}+P_{TE}(s)\log\frac{P_{TE}(s)}{P_{TR}(s)})$$

$$d_{\max}(PFSA_{TR},PFSA_{TE})=\max|P_{TR}(s)-P_{TE}(s)|$$

5. N-gram Model

An N-gram is a factor of fixed length N of a given behavior. Forrest & al. ([6], [14]) use the set of N-grams of the program behaviors as the program profile.

Let $H_{TR}=\{S_1, S_2,\ldots, S_L\}$ be the set of N-grams with L factors for TR, and let $H_b=\{ s_1, s_2,\ldots, s_l\}$ be the set of N-grams with l factors for a behavior in TE. Then the distance from s_i to H_{TR} is defined using the Hamming distance:

$$d(s_i,H_{TR})=\frac{1}{N}\min\{d_{Ham}(s_i,S_1),d_{Ham}(s_i,S_2),\ldots,d_{Ham}(s_i,S_L)\}$$

The distance between H_b and TR is the maximum distance from the factor in H_b to H_{TR}:

$$d(H_b,TR)=\max\{d(s_{i,}H_{TR})\}$$

6. Frequency-based Program Behavior Detection Method

Frequency-based methods take into account statistical analysis. Let $H_{TR}=\{S_1, S_2,\ldots, S_L\}$ be the set of N-grams for TR and $H_b=\{s_1, s_2,\ldots, s_l\}$ the set of N-grams for an observed behavior b in TE. Let NUM_{si} be the number of factor s_i appearing in TR and let num_{si} be the corresponding value for b. The frequencies of s_i in TR (noted F_{si}) and in b_{TE} (noted f_{si}) are defined by:

$$F_{s_i}=\frac{NUM_{s_i}}{\sum_{j=1}^{L}NUM_{s_j}},\quad f_{s_i}=\frac{num_{s_i}}{\sum_{j=1}^{l}num_{s_j}}$$

The distance between s_i and H_{TR} is defined as follows:

$$d(s_i,H_{TR})=|F_{s_i}-f_{s_i}|$$

The distance between b and TR is defined using max distance:

$$d(b,TR)=\max\{d(s_{i,}H_{TR})\}$$

IV. Combination Approach

In this section, we explain how to build a unique combined distance from different distances (associated to different methods). First, we present the Pearson correlation measure, which is a basic tool in the combinational approach.

A. Pearson Correlation Coefficients

Correlation indicates the magnitude and the direction of a linear relationship between two random variables. The correlation of two variables indicates how the two variables interact with each other. Pearson correlation coefficients are one of the basic correlation methods defined as follows:

Let $D_i(d_i(1),\ldots,d_i(n))$ and $D_j(d_j(1),\ldots,d_j(n))$ be two distances respectively for methods i and j. Their Pearson correlation coefficient is the ratio of the covariance of the two vectors by the product of their standard deviations.

$$Corr_{D_iD_j} = \frac{\text{cov } (D_i, D_j)}{\sigma_{D_i}\sigma_{D_j}} = \frac{\sum_{k=1}^{n}(d_i(k)-\overline{d_i})(d_j(k)-\overline{d_j})}{\sqrt{\sum_{k=1}^{n}(d_i(k)-\overline{d_i})^2}\sqrt{\sum_{k=1}^{n}d_j(k)-\overline{d_j})^2}} \qquad (1)$$

where $\overline{d_i} = \frac{1}{n}\sum_{k=1}^{n} d_i(k), \quad \overline{d_j} = \frac{1}{n}\sum_{k=1}^{n} d_j(k)$.

The value of Pearson correlation coefficients ranges from +1 to –1. They indicate the degree of linear dependence between the two distances. The closer the coefficient is to 1 or -1, the more closely they are related. If the coefficient is close to zero, it means that there is no relationship between the two distance variables. A positive coefficient means that the two distances evolve in the same way, a negative coefficient in different ways.

B. Rank Values of Methods

On a given data set, different methods have different performances. For giving to the best methods more weight in the combination, a rank value is associated to each method. The rank value of a method is a real number which is greater than 1. The rank of a method is computed during the process of training. We shall explain how in section 5.

C. Pearson Correlation Coefficients-Rank Matrix of Multi-distances

Suppose there are n distances. The Pearson correlation coefficient-rank (PCC-R) matrix M of n distances, is a nxn matrix. Let Rank(i) be the rank values of the ith method. By definition:

$$M(i,j) = \begin{cases} Corr_{D_iD_j}, & \text{if i} \neq \text{j} \\ Rank(i), & \text{if i=j.} \end{cases}$$

D. Unique Combined Distance

Given the Pearson correlation coefficient-rank matrix *M* and a distance vector ($d_1, d_2,\ldots,d_m$), where d_i is the corresponding distance of the i^{th} method, a combined distance is computed using the formula given in (2).

$$d_{combine} = (\alpha+1)\sqrt{(d_1, d_2, \ldots, d_m)M^{-1}(d_1, d_2, \ldots, d_m)^T} \qquad (2)$$

where M^{-1} is the inverse of the matrix *M*, $(d_1, d_2, \ldots, d_m)^T$ the transpose of the distance vector ($d_1, d_2, \ldots, d_m$) and a α nonnegative number which is chosen in the experiments.

E. The General Model Based on Combination

The method consists in six general steps.

(1) The parameters used for computing the distances associated to the different basic methods are extracted form the training and the test sets.
(2) The different distance values are computed.
(3) The PCC-R matrix is built using the distance vectors and the ranks.
(4) The values for the combined distance are computed.
(5) The threshold is computed.
(6) The distance of the tested behavior to the training data set are compared to the threshold.

V. Experiments

We have realized a simulation in C++ for validating the combination approach. The simulations concern two data sets: one is a data set of user behaviors and the other a data set of program behaviors. Both are public and experimentation results have been published for both of them.

A. Experiment with Data Set of User Behaviors

1. Data Set

The first data set collected by Schonlau & al. is used for masquerade detection [12]. The data include command traces, which come from the UNIX account audit mechanism. The data set provides 15,000 commands (150 blocks of 100 commands each) for each of 70 users. 50 users are selected randomly to serve as intrusion targets. Then the other 20 users are used as masquerade. For the 50 users, the first 50 blocks (5,000 commands) are used as training data. From blocks 51 to 150, the 20 users' data are shuffled into the 50 users' data, which are used as test data. The data are available for downloading from http://www.schonlau.net/intrusion.html (see [12] for more details of this procedure).

2. Experiment Design

Four methods – Uniqueness, Bayes one-step Markov, Naive Bayes, and Probabilistic finite state automata – are combined.

i. Training

During the training period, the Pearson correlation coefficient-rank (PCC-R) matrix is built. For the sake of simplicity, we explain how the coefficients are computed in the case of only two methods m and m'. There are two steps:

(1) For each user u, the commands from the 51th command through the 4950th command in training data are extracted and divided in blocks of 100 commands to obtain 49 cross blocks $\{u_{C151\sim C250}, \ldots, u_{C4851\sim C4950}\}$.
(2) For a given method *m*, the distances between each cross block with the training data are computed, yielding a distance vector $D_m(u)=\{D_m(u_{C51\sim C150}), \ldots, D_m(u_{C4851\sim C4950})\}$. Given two distance vectors $D_m(u)$ and $D_{m'}(u)$, corresponding to method *m* and *m'*, the Pearson correlation coefficients are computed by (1).

Now we present how the rank value of each method is computed. For each tested user, we consider other legitimate users as masqueraders to estimate the efficiency of each combined method and to determine the corresponding rank value. The rank value of each combined method is initially 1. Then it increases each time a new legitimate user is added in the process of training. The rank value of a method for a particular user is computed as follows. Let $D_m^{max}(u)$ be the maximum distance in the distance vector $D_m(u)=\{D_m(u_{C51\sim C150}), \ldots, D_m(u_{C4851\sim C4950})\}$ of method *m* for user *u*, let $d_{uu'}(m)$ be the distance between the two training data of users *u* and *u'* in method *m*, let N_m be the number of combined methods (here 4), and let $N_{u'}$ be the number of users that is chosen as masqueraders (here 49). The rank value of the method *m* for user *u* is:

$$Rank_u(m) = 1 + \frac{1}{N_{u'}} \sum_{u'} \frac{1}{\frac{d_{uu'}(m)}{D_m^{\max}(u)} + \frac{1}{N_m}} \quad (3)$$

The ratio of $d_{uu'}(m)$ and $D_m^{\max}(u)$ indicates the ability for the method *m* to distinguish between the behaviors of users *u* and *u'*. The larger the value of the ratio is, the better the method makes a distinction between users *u* and *u*'. In the same way, the larger the value of the rank is, the worse is the performance of the method. $\frac{1}{N_m}$ is added to avoid a null denominator.

ii. Test

During the test period, the combination method decides whether or not a block in the test data has to be considered as a masquerade. First, for each method the distance between the test block and each block in the training data are computed. The distance between the test block and the training data is the minimal distance for all blocks in the training data. Then the formula (2) is used to combine these distances. In this experiment, α is the number of methods deciding that the test data is an intrusion.

3. Threshold Determination

The value of the threshold directly influences the false and missing alarm rates. In this experiment, we determine an individual threshold for each user instead of fixing the value for all of them. For each user, two types of thresholds are computed: the threshold of each method and the global threshold of the combination of these methods.

i. Thresholds for Individual Method

Recall that $D_m^{max}(u)$ is the maximum distance of $D_m(u)=\{D_m(u_{C151\sim C250}),\ldots,D_m(u_{C4851\sim C4950})\}$. Let $d_{uu'}(\max)$ and $\overline{d_{uu'}}$ be the maximum distance and the mean distance from other legitimate users to the tested legitimate user.

Let $D_{combine}(u)= \{D_{combine}(u_{C151\sim C250}),\ldots, D_{combine}(u_{C4851\sim C4950})\}$ be the vector whose elements are the combined distances of the cross blocks to the training data. Let σ_u be the standard deviation of $D_{combine}(u)$, and let $\overline{\sigma_u}$ be the mean of all of σ_u. σ_u can be considered as an indicator of the regularity of user behaviors. The larger σ_u is, the less regular user *u*'s behaviors are. The threshold of method *m* for user *u* is defined by:

$$TS_m(u) = \begin{cases} (D_m^{\max}(u)+\overline{d_{uu'}})/2, & \text{if } \sigma_u \le 1.5\overline{\sigma_u} \\ (D_m^{\max}(u)+d_{uu'}(\max))/2, & \text{if } \sigma_u > 1.5\overline{\sigma_u} \end{cases}$$

ii. Global Threshold

Recall that M^{-1} is the inverse of the Pearson correlation coefficient-rank matrix and $(TS_1,TS_2,\ldots,TS_{Nm})^T$ is the transpose of $(TS_1,TS_2,\ldots,TS_{Nm})$. The global threshold for user *u* is computed as follows:

$$threshold_{global}(u) = \beta\sqrt{(TS_1,TS_2,\ldots,TS_{N_m})M^{-1}(TS_1,TS_2,\ldots,TS_{N_m})^T}$$

Where β is a real number greater than 0. The false alarm and missing alarm rates depends on the chosen value for β.

4. Update

Our method has an aging version, in which the training set evolves dynamically and the ranks and thresholds are computed from the new training data set regularly. The idea for updating the training data set is that, when a particular test block is considered as normal by all of methods and its combined distance is less than the maximum combined distance of $D_{combine}(u)$, this block is added to the training data.

5. Results

The quality of an IDS can be measured by the probabilities of false and missing alarms. We compare the false alarm rates and the missing alarm rates of our method with 9 other methods in Table I. The PCC-R combination is the combination approach using Pearson correlation coefficients-rank matrix using the global threshold with β equal to 2.

TABLE I
COMPARISON WITH FALSE ALARM AND MISSING ALARM RATES FOR THE MASQUERADE DETECTION METHODS

Approach	Missing Alarm Rate	False Alarm Rate
Compression[12]	65.8%	5.0%
Sequence-match[12]	63.2%	3.7%
Uniqueness[12]	60.6%	1.4%
IPAM[12]	58.9%	2.7%
Hybrid Multistep Markov [12]	50.7%	3.2%
Naive Bayes (updating)[9]	38.5%	1.3%
Naive Bayes (not updating) [9]	33.8%	4.6%
Bayes 1-step Markov[12]	30.7%	6.7%
Semi-Global Alignement [1]	24.2%	7.7%
PCC-R combination (not updating)	**18.7%**	**4.9%**
PCC-R combination (updating)	**21.1%**	**3.2%**

Missing alarms: In current methods, the average of missing alarm rate is about 40% (from 25% to 65%). It appears that the PCC-R compositional approaches have the best results. In case of not updating, the missing alarm rate we obtain is about 22% lower than the average value of the current methods, and about 6% lower than the best one (Semi-Global Alignment). Compared to the Naive Bayes (not updating) method, which has about the same performance for false alarm rate, the PCC-R combination (not updating) has about two times less missing alarms.

False alarms: In current methods, the false alarm rates go from 1.4% to 7.7%. The value for the PCC-R combination is on the average. The results of the combination method are better than the results of the two other best methods for missing alarm rate, Bayes 1-step method and Semi-Global Alignment method.

TABLE II
COMPARING DETECTION OF ANOMALIES

	N-grams	Frequency	PFSA	PCC-R Combination
sunsendmail	0.636	0.138	8.443	33.04
decode	0.2	0.031	1.726	7.554
forwardloops	0.545	0.08	6.025	23.161

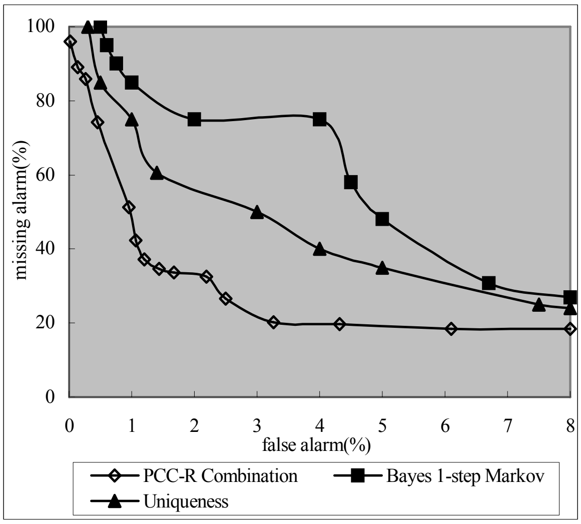


Fig. 3 Comparison with two combined methods

Fig. 3 represents the value of the missing alarm rate in function of the false alarm rate (each point corresponding to a given value for the threshold). Roughly speaking the faster a curve decreases, the better is the method. In most practical environment, too many false alarms will make the system useless. The comparison of the detection rate with a small false positive rate is more interesting. Therefore, we only show the portion of the curve where the false positive rate is smaller than 8% in Fig. 3. We compare our results with two other methods: Bayes one step Markov method and Uniqueness method. We can see that our curve is under the other curves. It means that our compositional approach always detects more masquerades than others.

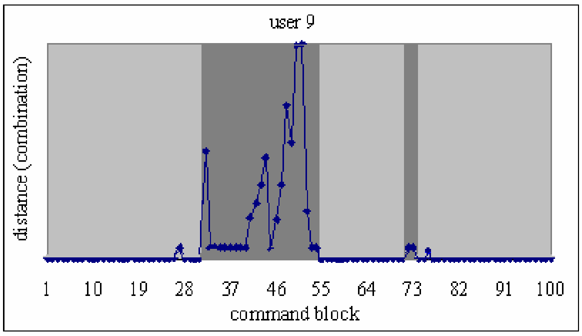


Fig. 4 Distances of all of the test data for a user

Fig. 4 gives the distances of all 100 blocks of test data for the user 9 as an example of how our method distinguishes the masquerade blocks. The user's test data include long masquerade sequences (24 blocks), displayed on a dark backgrounds. The line indicates the distance for our combination method of the corresponding block of test data. It can be seen that our model clearly separates most of the masquerades data from the normal data.

B. *Experiment with Data Set of Program Behaviors*

1. Data set

To facilitate the comparison with other methods based on program behaviors, we consider the data set of UNIX system calls collected by Forrest & al. [6]. The data set includes different kinds of programs and different kinds of intrusions (buffer overflows, symbolic link attacks, Trojan programs and DOS). Each trace is a sequence of system calls issued by a single process from the beginning of its execution to its end. The data set can be downloaded at

http://www.cs.unm.edu/~immsec/data-sets/html.

Synthetic sendmail programs are tested in the experiment. The sets of training data and test data are summarized as follows:

· Training data set: a trace of the sendmail daemon and 4 other traces under various normal conditions.

· Test data set: the normal traces that are not used in the training data set, a trace of sunsendmail attack (sm-10763), a trace of decode attack (sm-280), a trace of forward loops attack (fwd-loops-1)

2. Experiment Design

Three IDS methods are used here: N-gram method, Frequency-based method and Probabilistic finite state automata (PFSA).

i. Training

In the same way as for the first experiment to compute the ranks, we choose randomly a trace in the normal data set of the sendmail program and the traces in the normal data of other programs (i.e. ps program, login program, etc.). The detailed presentation is in the last section.

To compute the Pearson correlation coefficients, a distance vector is constructed for each method (see (1) in section 4). We present now how the distance vectors are built. Since the training data includes a large number of system calls (there are more than 1.5 million system calls in the training data of the sendmail program), computing the distances between the training data and the cross blocks in training data as we did in the first experiments, produces too much elements for each vector. To reduce the complexity some traces in normal data of other programs are chosen randomly, and their distances to the training data are computed, yielding the distance vector.

ii. Test

For being able to perform online detection, each trace of test data is divided into continuous blocks. Each block groups 100 system calls.

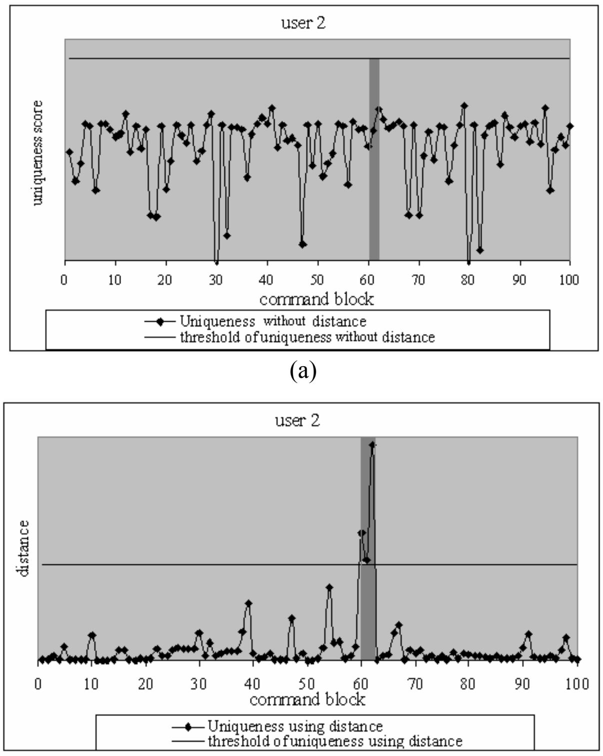
A difference with the first experiment is that the intrusion detection is made on an entire trace rather than on blocks. For each method, the blocks of test data are compared with the training data one by one, and the distance between the test trace and the training data is the mean of distances from the test blocks to the training data. Additionally, α in formula 6 is chosen equal to 0.

3. Results

Intrusion detection for program behaviors is easier than for user behaviors, because program behaviors are always more regular than user behaviors. The false alarm and missing alarm rates are often not enough precise. That is the reason why we choose as an indicator the distance of the test trace to the training data set and more precisely the difference between the distance between the intrusion data and the training data and the distance between the normal data and the training data. The values of this difference are represented in Table II. The larger this difference is, the more easily intrusions are detected.

C. *Comparison with Distance and without Distance*

In Fig. 5, we show that making the distance explicit yields better results. Fig. 5(a) displays the original results of the Uniqueness method (without explicit distance). The horizontal line is the threshold. A dark background corresponds to masquerade blocks. Fig. 5(b) presents the result on the same data when the Manhattan distance is used with the same method. It can be seen that the use of distance separates the intrusive and normal behaviors more efficiently.



(a)

(b)

Fig. 5 Comparison of the results with distance and without distance

D. *Comparison with Other Combination Approaches*

In this section, we compare the PCC-R method with two other combination approaches: sequential-based and majority-based.

The idea of the sequential-based approach is that the methods are applied one after the other. An alarm is raised if and only if each method considers an observed behavior as abnormal.

Majority-based approach can be considered as a voting system. If a majority of methods consider an observed behavior as intrusive, an alarm is raised.

We compare the results of sequence-based combination and majority-based combination with the PCC-R combination approach in Table III using the data set of the masquerade detection (the data set that is used in the first experiment).

TABLE III
COMPARISON WITH OTHER COMBINATION APPROACH

Combination Approaches	Missing Alarm Rate	False Alarm Rate
Sequential-based	63.6%	0.6%
Majority-based	41.0%	2.0%
PCC-R	21.1%	3.2%

Concerning the sequential-based combination, the threshold of each method for a user is the maximum distance between the behaviors in the user's training data. The majority-based combination uses the same thresholds with the PCC-R combination that we have presented.

The advantage of the sequential-based combination is that it reduces greatly the false alarm rate. The false alarm rate is under 1%. Compared with the uniqueness method, it detects the same number of intrusion, while its false alarm rate is 2 times lower than for uniqueness method. But the sequential-based combination approach misses a majority of intrusions, which is not acceptable in practice.

Compared with the sequential-based combination, the majority-based combination has a better trade-off between false alarm and missing alarm rates. However in comparison with the PCC-R combination, it detects 20% less masquerades than the PCC-R combination, with only 1% less false alarms.

E. *Discussion*

According to the results, the combination method decreases greatly the missing alarm rate (see Table I) and improve the distinction between normal and intrusion data (see Table II). Table I shows that the missing alarm rate for the PCC-R combination method is about 18%, which is at least 10% lower than the best among the other existing methods. In addition, Table II shows that the use of combination improves the difference between the normal and intrusion data at least about 4 times better than the methods that it combines.

One can think of three reasons for explaining the better performances:

- The combination model considers an observed behavior from several points of views, increasing the capacity of intrusion detection.

· The distance allows to measure the dissimilarities between the normal behaviors and the abnormal behaviors.

· The Pearson correlation coefficient-rank matrix expresses the weight of the method statistically, combining the advantages of each of them.

VI. Conclusion

In this paper we present a unified model able to combine different methods and gain advantage of each of them. The idea is to associate a distance with each method and use the Pearson correlation coefficient-rank matrix to combine these different distances. A distance measures the dissimilarity between two behaviors and allows distinguishing the normal from the abnormal behaviors. The experiments prove that the combination model is able to improve some known results. We made two experiments based on two different data sets: a user behavior data set and a program behavior data set.

The first experiment shows that the combination model detects much more intrusions with just a little more false alarm rate.

The second experiments shows that the combination model makes easier the distinction between the intrusion and the normal data.

The fact that the combination method has better results for two different data sets suggests that the improvement comes from the method itself.

References

[1] S. Couil, J. Branche, and B. Szymanski, "Intrusion Detection: A Bioinformatics Approach," in *Proc. 19th Annu. Computer Security Applications Conf,* Las Vegas, Nevada, Dec. 2003.

[2] K. Christopher, V. Giovanni, "Anomaly detection of web-based attacks," in *Proc. 10th ACM Conf. Computer and Communications Security*, Wanshington D.C., USA, Oct. 2003. ACM Press New York, NY, USA.

[3] S. Forrest, S. Hofmeyr, A. Somayaji, T. Longstaff, "A sense of Self For Unix Processes," in *Proc. 1996 IEEE Symposium on Security and Privacy*, Oakland, California, USA, May 1996, pp.120-128. IEEE Computer Society Press, Los Alamitos, California.

[4] S. Freeman, "Host-based Intrusion Detection Using signatures," in *Graduate Research Conf.* Troy, NY, 2002.

[5] H.S. Javitz, A. Valdes, "The SRI IDES statistical anomaly detector," in *Proc. 1996 IEEE Symposium on Security and Privacy*, Oakland, California, USA, May 1991, pp.316-326. IEEE Computer Society Press, Los Alamitos, California.

[6] W. Lee, S.J. Stolfo, " A framework for constructing features and models for intrusion detection systems," *ACM Trans. Information and system security*, vol.3, no. 4 , 2000, pp.227-261.

[7] W. Lee, S.J. Stolfo, "Data Mining Approaches for Intrusion Detection," in *Proc. 7th USENIX Security Symposium*, San Antonio, Texas, January 1998, pp.26-29.

[8] D.E. Denning, "An intrusion-detection model," *IEEE Trans. Software Engineering*, vol.13, no. 2 , Feb. 1987, pp. 222-232.

[9] R. Maxion, T. Townsend, "Masquerade Detection Using Truncated Command Lines," in *Int. conf. on Dependable Systems and Networks*, Washington, D.C., American, June 2002 pp. 219-228. IEEE Computer Society Press, Los Alamitos, California.

[10] D. Gao, M. K. Retier, D. Song, "Behavioral distance measurement using hidden markov models", In *Conf. Recent Advanced in Intrusion Detection (RAID)*, Hamburg, Germany, Sep. 2006, pp.19-40.

[11] S. Rubin, S. Jha, B. Miller, "Automatic generation and analysis of NIDS attacks," in *proc. 20th Annu. Computer security applications conf.* Tucson, AZ, USA, Dec 2004, pp 28-38. IEEE Computer society 2004.

[12] M.Schonlau, W.DuMouchel, "Computer Intrusion: Detecting Masquerades," *J. Statistical Science*, vol.16, no.1, Feb 2001, pp. 58-74.

[13] M. Srinivas, H.S. Andrew, A. Ajith, "Intrusion detection using an ensemble of intelligent paradigms," *J. nerwork and computer applications*, vol 28, 2005, pp. 167-182.

[14] A. Steven, S. Hofmeyr, S. Forrest, and A. Somayaji, "Intrusion Detection using sequences of system calls," *J. Computer Security*, vol. 6, no. 3 1998, pp. 151-180.

[15] C. Warrender, S. Forrest, B. Pearlmutter, "Detecting intrusions using system calls: alternative data models," In *Proce. 1999 IEEE Symposium on Security and Privacy*, Oakland, California, USA, May 1999, pp.133-145. IEEE Computer Society Press, Los Alamitos, California.

[16] W. Fan, S. Stolfo, "Ensemble-based adaptive intrusion detection", In *Proc. SIAM Inter. Conf. Data minging 2002*.

[17] F. Gianluigi, P. Clara, S. Giandomenico, "GP ensemble for distributed intrusion detection systems", *Pattern Recognition and Data Mining*, vol 3868, pp.54-62, Sep. 2005.

[18] G.Giacinto, F. Roli, "Intrusion detection in computer networks by multiple classifer systems", In *Proc. 16th Inter. Conf Pattern recognition.*, Quebec, Canada, 2002, pp.390-393.

Joffroy Beauquier is full Professor at Université de Paris-Sud XI (France) where he teaches distributed algorithms and security. He formerly studied formal language theory, concurrency, and fault-tolerant distributed systems. He now works on intrusion detection. He is the author of more than one hundred scientific papers and of a teaching book (in French) on operating system principles.

Yongjie Hu is PHD student at Université de Paris-Sud XI (France). The subject of her PHD research is about intrusion detection and tolerance system. She has published several scientific papers in the journals.