Serdica J. Computing **3** (2009), 335–358

Serdica
Journal of Computing

Bulgarian Academy of Sciences
Institute of Mathematics and Informatics

CLASSIFICATION TREES AS A TECHNIQUE FOR CREATING ANOMALY-BASED INTRUSION DETECTION SYSTEMS

Veselina Jecheva, Evgeniya Nikolova

Abstract. Intrusion detection is a critical component of security information systems. The intrusion detection process attempts to detect malicious attacks by examining various data collected during processes on the protected system. This paper examines the anomaly-based intrusion detection based on sequences of system calls. The point is to construct a model that describes normal or acceptable system activity using the classification trees approach. The created database is utilized as a basis for distinguishing the intrusive activity from the legal one using string metric algorithms. The major results of the implemented simulation experiments are presented and discussed as well.

I. Introduction. Intrusion detection systems are essential parts of the contemporary security systems, which are aimed at protecting various kinds of networks, from simple home networks to multinational commercial networks.

ACM Computing Classification System (1998): C.2.0.
Key words: Intrusion detection, Data mining, String metrics, Similarity coefficients.

Their role is to monitor the computer and network systems with the purpose of detecting any violations of the accepted security policy.

Intrusion Detection Systems (IDS) monitor system behaviour and alert on potentially malicious network traffic [3]. They were categorized, based on their detection models, into the following: misuse detection model and anomaly detection model. Misuse-based IDS looks for signatures of common attacks in the current system data and alerts the activities accordingly [4], similarly to most anti-virus products. They produce reliable results and a low level of false attacks. The major disadvantage of these systems is their inability to detect new attacks or variations of common attacks, since they discover only intrusions that match previously known attack scenarios.

In contrast, anomaly detection approaches build models of normal data and then attempt to detect deviations from the normal model in observed data [26]. If any significant deviation is found, the IDS raise an alarm in the case the event is classified as an attack. The anomaly-based IDS have the capability to detect novel attacks and intrusions without known signatures, as they attempt to search for malicious behaviour that deviates from established normal patterns [22]. However, describing the normal activity and the deviation from it is not a trivial task. Therefore, anomaly-based IDS produce higher false alarms compared to misuse-based approaches and could miss real attacks because of a deficiency in their ability to discriminate attacks from legitimate behaviours [24].

Until now, to model normal and abnormal system behaviours using intrusion audit data, various techniques have been applied: data mining [6, 25], Hidden Markov Model [2, 23], fuzzy logic, genetic algorithms [8, 16, 17], neural networks [7, 35], etc.

Among the most critical issues of the IDS is to profile normal behaviour at a level that is both robust to variations in normal and perturbed by intrusions [13]. The different methods for anomaly detection vary in how they describe normal activity and how they define deviation from this baseline. Many anomaly detection approaches [9, 30, 37] define normal behaviour using a run-time process activity. System call traces are a common type of audit data collected for performing intrusion detection. A system call trace is the ordered sequence of system calls that a process performs during its execution [34]. This supervised approach includes two major stages. The first phase involves collecting traces of normal behaviors and building a database to characterize normal patterns from the observed system calls. In the second phase, newly observed system call sequences are matched against the normal pattern of the system behaviour [10].

Forrest and Ghosh [12, 13, 14] propose an approach that monitors the

system behaviour at the level of running services (for example, ftp, sending or receiving e-mail, etc.), which they referred to as privileged processes. In order to be able to perform their functions, these processes are given much more privileges in the system, compared to those of the ordinary users. For that purpose, they are special targets for the attackers, since a successful intrusion would give them much more control over the system compared to the attacks against the programs, which are invoked by the ordinary users. Monitoring privileged processes also offers some advantages over monitoring user behaviour, since the behaviour of the privileged processes varies to a lesser extent compared to the behaviour of the users, which can involve diverse actions. The functions performed by the privileged processes rarely change over time.

The present work addresses the issue of anomaly-based IDS, which describes and monitors the behaviour of privileged processes, based on the system call sequences. Since a complete set of system call sequences can hardly be built in real environments, the proposed methodology applies some data-mining techniques for description of the normal system activity. The detection of abnormal process behaviour during the system work is performed using some similarity coefficients and distance measures, which are widely used in various recognition tasks.

II. Similarity and distance measures. There are two major techniques – *similarity measures* and *distance measures*, which could be applied with the purpose of determining the degree of similarity between two sequences. The similarity measure gives a quantitative value which is higher in the case of greater similarity. Conversely, a greater value of the distance measure indicates a lesser similarity.

The aim of this paper is to determine how the selection of a similarity coefficient affects the resulting classifications when measuring the similarity of two sequences of system calls. The results showed that for almost all methodologies and marker systems, the Jaccard, Sorensen-Dice and Anderberg coefficient showed close results.

1. Wagner-Fischer distance, Jaro distance and Jaro-Winkler distance. In this section our attention is focused on the Wagner-Fischer distance (WFD) [36], the Jaro distance (JD) [20] and the Jaro-Winkler distance (JWD) [38]. The WFD is a string metric between two strings, which stands for the minimum number of operations (insertion, deletion, substitution of a single character, transposition of two characters) needed to transform one string into the other. Let the weight for the cost of transforming symbol a into symbol b be denoted by

$w(a,b)$. Then $w(a,b)$ is the cost of a symbol substitution $a \to b$, $w(a,\varepsilon)$ is the cost of deleting a and $w(\varepsilon,b)$ is the cost of inserting b. The *WFD* are computed using the following recurrence relation:

$$d_{WF}(i,j) = \min \left\{ \begin{array}{l} d(i-1,j) + w(x_i,\varepsilon), d(i,j-1) + w(\varepsilon,y_j), \\ d(i-1,j-1) + w(x_i,y_j) \end{array} \right\}.$$

It calculates the exact number of operations needed to transform the string into the other one. The distance between two strings is zero if they are identical. This value is referred to as *restricted edit distance*.

The *JD* is a measure of similarity between two strings without being a metric in the mathematical sense of that term. Given two strings s_1 and s_2, their *JD* is

$$d_J = \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right),$$

where

- m is the number of *matching* characters (s_1 and s_2 are *matching* only if they are not farther than $\left\lfloor \frac{\max(|s_1|,|s_2|)}{2} \right\rfloor - 1$);
- t is the number of *transpositions* (the number of matching characters divided by two).

The *JWD* are computed using the following formula:

$$d_{JW} = \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) + lp\left[1 - \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right)\right],$$

where

- l is the length of the common prefix at the start of the string;
- p is a constant scaling factor for the degree of closeness for having common prefixes.

2. Similarity measures for sequences. *Similarity* $s(p,q)$ is a numerical measure of how alike two data sequences are. It gets values in the interval [0.1]. Let $p$and $q$are the attribute values for two data sequences, then

$$s(p,q) = \left\{ \begin{array}{lll} 1 & \text{if} & p = q \\ 0 & \text{if} & p \neq q \end{array} \right. .$$

The similarity is symmetric, i.e. $s(p,q) = s(q,p)$ for all p and q.

If a sequence x is defined as a concatenation of symbols from a finite alphabet Σ, a language $E \in \Sigma^*$ comprises subsequences $\omega \in E$, which are called *words*. Given a language E, a sequence x can be mapped into an $|E|$-dimensional space by calculating a function $\phi_\omega(x)$ for every $\omega \in E$ appearing in x. The function $\phi_\omega(x)$ is defined as follows

$$\phi_\omega : \Sigma^* \to \mathrm{R}^+ \cup \{0\}, \quad \phi_\omega(x) = \psi(occ(\omega, x)).\mathcal{W}_\omega$$

where $occ(\omega, x)$ is the number of occurrences of ω in x, ψ is a numerical transformation, e.g. a conversion to frequencies and $\mathcal{W}_\omega$ is a position-depended weight.

Special attention is drawn to the following similarity coefficients for sequential data [1, 5, 19, 21, 31]:

Jaccard (J) $$s_J(x,y) = \frac{a}{a+b+c}$$

Czekanowski-Sorensen-Dice (CSD) $$s_{CSD}(x,y) = \frac{2a}{2a+b+c}$$

Sokal-Sneath-Anderberg (SSA) $$s_{SSA}(x,y) = \frac{a}{a+2(b+c)}$$

where

$$a = \sum_{\omega \in L} \min\left(\phi_\omega(x), \phi_\omega(y)\right)$$

$$b = \sum_{\omega \in L} \left[\phi_\omega(x) - \min\left(\phi_\omega(x), \phi_\omega(y)\right)\right]$$

$$c = \sum_{\omega \in L} \left[\phi_\omega(y) - \min\left(\phi_\omega(x), \phi_\omega(y)\right)\right]$$

III. Description of the methodology. Data mining in general is the process of discovering useful and previously unknown information from historical or real-time data [18]. Data mining in intrusion detection may vary from the simple task of determining the relationships among a set of host or network data to modelling certain tasks, such as attacks classification and accordingly a response choice. The data mining process involves several consecutive steps, namely:

- data pre-processing, which involves preparation, selection and more importantly understanding data characteristics;

- data analysis, which is essentially a search for useful patterns of any form in the data;

- the process of intrusion detection itself, which includes scanning of real-time or archived data collected during the system work.

Classification tree is another frequently applied approach in the field of intrusion detection [15, 27]. In our approach the implementation of the classification trees is performed through the process of description of the normal system activity. The normal activity patterns compose a set Q with N states $q_1, q_2, \ldots, q_N$, which the system passes through its work in the discrete moments of time $t = 1, \ldots, T$. We assume that the probability of occupying a state is determined solely by the preceding state. Each state transition probability represents the probability of transitioning from a given state to another possible state. Based on the state transition probabilities, we construct classification trees of level L, whose roots are all possible states q_k, $k = 1, \ldots, N$. The inheritors for each vertex are the states for which the corresponding transition probabilities from their predecessor are non-zero.

By traversing the tree from the root to the leaves we can obtain all possible state sequences with length L along with the corresponding transition probabilities. The obtained lists of system calls consist of all possible sequences with a given state in k^{th} position and contain states for which the transition probabilities for each couple of neighbours is non-zero.

Within the created classification trees we apply the *WFD*, *JD*, *JWD*, *SSA* similarity coefficient, *CSD* similarity coefficient and *JD* similarity coefficient between the obtained sequence and normal sequences for the calculation of the number of errors. The obtained value of 1 means that the observed sequence contains no intrusions, while the value of 0 stands for a sequence of intrusions only. Since the similarity coefficients report only the degree of proximity, for the assessment of abnormal positions we apply the Hamming distance between the received sequence and the list $\boldsymbol{S}_i$, consisting of the sequences with the same maximal similarity coefficient:

$$d_H = \sum_{j=1}^{n} \delta_j, \quad \delta_j = \begin{cases} 1, & s_j^i = y_j \\ 0, & s_j^i \neq y_j \end{cases}, \quad \boldsymbol{S}_i = (s_1^i, \ldots, s_n^i), \quad y = (y_1, \ldots, y_n).$$

IV. Statistical methods of evaluating the effectiveness of IDS. The goal of our classification test is to determine whether a given sequence belongs to one of two sets—the normal set or the intrusion set. For every possible test value there are two kinds of errors—*a false positive* (FP) and *a false negative* (FN). A false positive occurs when an event is predicted as intrusive but it is in fact normal. A false negative occurs when a truly intrusive event occurs without being signalled. If the target value is greater than the given threshold, the data is signalled as intrusive, and is considered as normal otherwise. We have a hit if a truly intrusive session is registered by a given test as intrusive. We can compute the hit rate (HR) as the ratio of number of the hits on the intrusive session to the total numbers of intrusive sessions in the testing data and false-alarm rate (FAR) as the ratio of the number of the false alarms to the total number of the truly normal data.

A *Receiver Operating Characteristic Curve* (ROC curve) [11] plots hit rates and false-alarm rates for various thresholds. The closer it is to the top left corner, with 100% hit rate and 0% false-alarm rate, of a chart, the better the performance. Hence, the ROC curve shows the overall detection performance of a given test.

As a measure of the quality of binary classification can be used the *Matthews correlation coefficient* (MCC) [28].

$$MCC = \frac{TP.TN - FP.FN}{\sqrt{(TP + FP)\,(TP + FN)\,(TN + FP)\,(TN + FN)}},$$

where TP is the number of true positives and TN is the number of true negatives. $MCC = +1$ represents a perfect prediction and $MCC = -1$ represents the worst possible prediction.

The performance of each classifier was evaluated using the detection rate and overall accuracy. The detection rate shows the percentage of the true intrusions that have been successfully detected. It is a function of the identified intrusions:

$$\text{Detection rate} = \frac{TP}{TP + FN}$$

The overall accuracy [33] is a percentage of correctly identified patterns:

$$\text{Overall Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

The false positive rate (FPR) is the frequency with which the IDS reports malicious activity in error.

$$\alpha = \text{False positive rate} = \frac{\text{number of false positives}}{\text{total number of negative instances}}$$

To facilitate performance comparison among different methods the cost function was used [29]:

$$\text{Cost} = (1 - \text{hit rate}) + \gamma.\text{false positive rate},$$

where the parameter γ represents the relative cost difference between a false positive rate and a miss. The lower the cost is, the better performance an intrusion detection system has.

Spearman's Rank correlation coefficient [32] is a technique which can be applied in order to summarise the strength and direction (negative or positive) of the relationship between two variables. Spearman's Rank correlation coefficient is given by the following formula

$$SRCC = 1 - \frac{6 \sum d_i^2}{n^3 - n},$$

where d_i is the difference between the ranks of corresponding values and n is the number of pairs of values. If the $SRCC$ value

- is -1, there is a perfect negative correlation;
- falls between -1 and -0.5, there is a strong negative correlation;
- falls between -0.5 and 0, there is a weak negative correlation;
- is 0, there is no correlation;
- falls between 0 and 0.5, there is a weak positive correlation;
- falls between 0.5 and 1, there is a strong positive correlation;
- is 1, there is a perfect positive correlation

between the two sets of data.

V. Simulation experiments and results.

1. Description of the simulation data. Extensive empirical testing of the proposed methodology was performed on the data generated and published by the Immune Systems Project of the Computer Science Department, University of New Mexico. The data are obtained from Unix system examination during an extended period of time and consist of normal user activity patterns of some

privileged processes executed on behalf of the root account as well as some anomalous data. The methods for pattern generation are described in [12] and [13]. They prove that the short sequences of system calls are a reliable discriminator between normal and anomalous activities in the system. Each pattern is a sequence of system calls, which are the results of the examined process. The input data files are sequences of ordered pairs of numbers, where each line consists of one pair. The first number in each pair is the process ID (PID) of the process executed, and the second one is the system call number. Forks are taken into account as separate processes and their execution results are considered as normal user activity.

As a first stage based on the normal user activity patterns, the state transition probabilities for the sequences of the normal system activity were evaluated and the normal database, which consists of the classification trees of level L, was created. These trees compose the normal program behavior profiles. During the second stage, which is the intrusion detection itself, the anomalous data were divided into portions of length L and compared to the lists extracted by the trees in normal database. The testing data contain both normal and anomalous patterns for the following processes: inetd, login, named and synthetic sendmail. The anomalous data for the processes login and named contain two separate files, designated in the results as login1, login2, named1 and named2, respectively.

2. Simulation resuts. The anomalous activity was detected using *WFD*, *JD* and *JWD*, as representatives of the string metrics, and *SSA*, *CSD* and *J* similarity coefficients. The distance distributions for the string metrics *WFD*, *JD* and *JWD*, which give us information about the number of anomalous

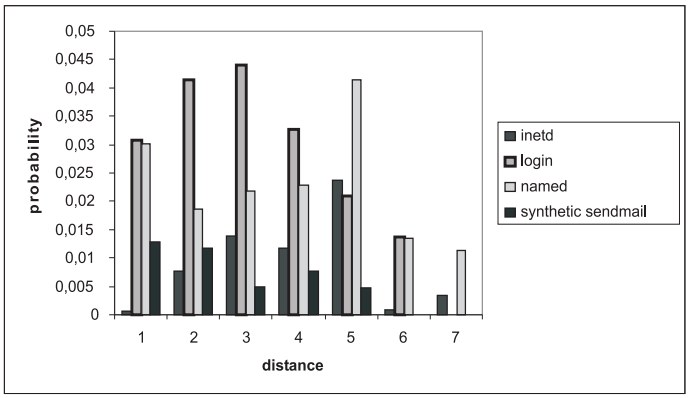

Fig. 1. The distances distributions of WFD

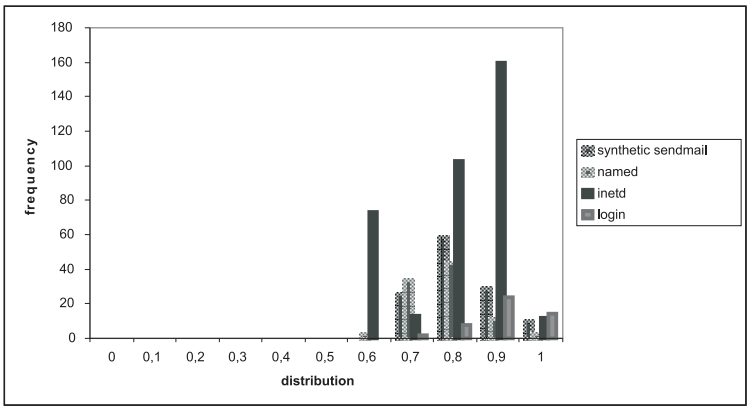

Fig. 2. The distance distributions of *JD*

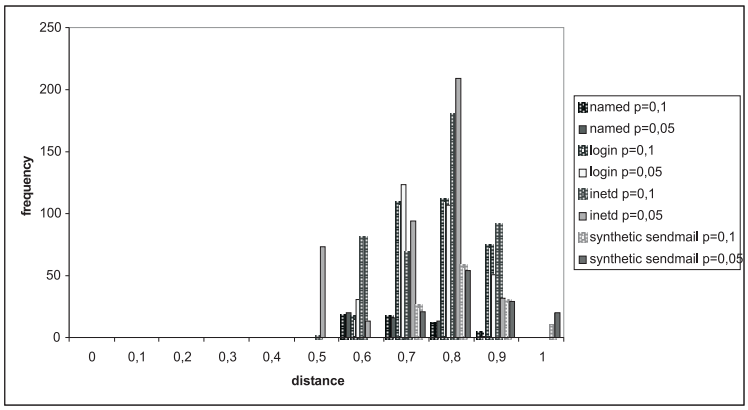

Fig. 3. Distance distributions for the processes inetd, login, named and synthetic sendmail using the IDS based on *JWD*

patterns in the examined sequences of length L, are represented in Figures 1–3 respectively. The calculated distances are numbers between 0, which indicates that the observed sequence is a result of normal activity, and L, which indicates that the observed sequence contains only attacks.

The distributions of the number of anomalous patterns in the examined sequences of observations with length $L = 7$ detected by IDS based on *JD* or *JWD* with scaling factor $p = 0.05$ and $p = 0.1$ are represented respectively in Figures 4 and 5.

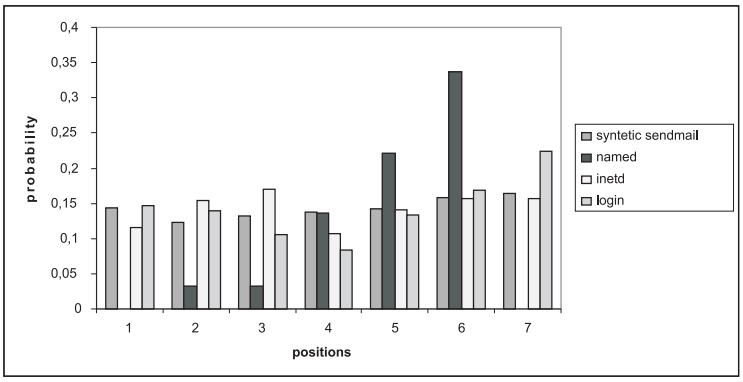

Fig. 4. Frequency of a number of intrusions in a sequence of observations with length $L = 7$, detected by IDS based on *JD*

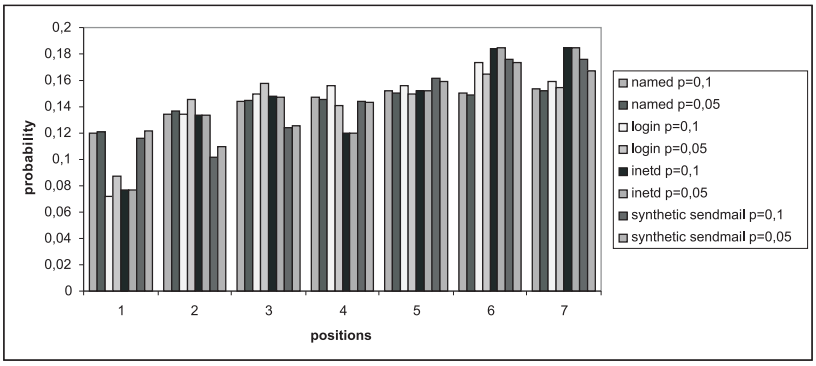

Fig. 5. Frequency of a number of intrusions in a sequence of observations with length $L = 7$, detected by IDS based on *JWD* with scaling factor $p = 0.05$ and $p = 0.1$

According to the methodology described in the previous section, we calculated the *SSA*, *CSD* and *J* similarity coefficients in order to evaluate the degree of similarity between the observed sequences and the sequences obtained by traversing the trees from the normal activity profiles. The calculated coefficients are numbers between 0, which indicates that the observed sequence contains only attacks, and 1, which indicates that the observed sequence is a result of normal activity. The result similarity coefficients for the processes named, synthetic sendmail, inetd and login, are represented in Figures 6–8. From the figures we can see that the coefficient distributions depend on the executed processes. In the case that more than one normal sequence with maximum similarity coefficient was obtained, the sequence with minimum Hamming distance was searched

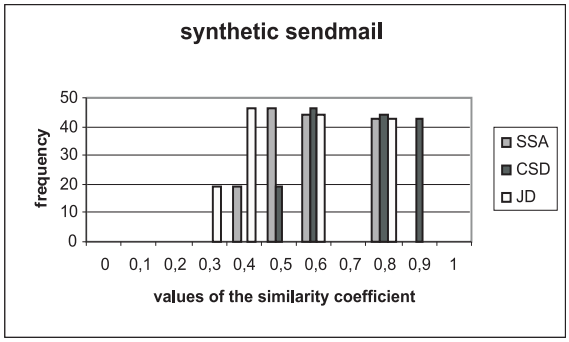

Fig. 6. The distance distributions of the *SSA*, *CSD* and *J* similarity coefficient for the process synthetic sendmail

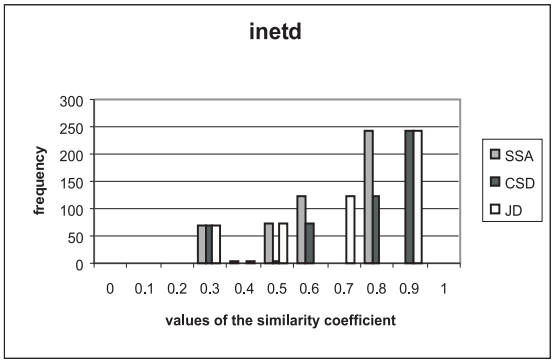

Fig. 7. The distance distributions of the *SSA*, *CSD* and *J* similarity coefficient for the process inetd

for. The relations between the similarity coefficients and the Hamming distances obtained for the sequences with equal coefficient value are represented in Figures 9–11.

3. Effectiveness of the applied methodology. The presented anomaly detection methods could accurately ascertain a given unknown sequence to be normal or anomalous with a detection rate whose values for the processes inetd, login, named and synthetic sendmail are presented in the Tables 1 and 2. One important question is whether the choice of the distance or similarity coefficient has a significant influence on the efficiency of the methodology. With the purpose of comparing the different intrusion detection methods, the Matthews correlation coefficient and overall accuracy were computed and described in section IV.

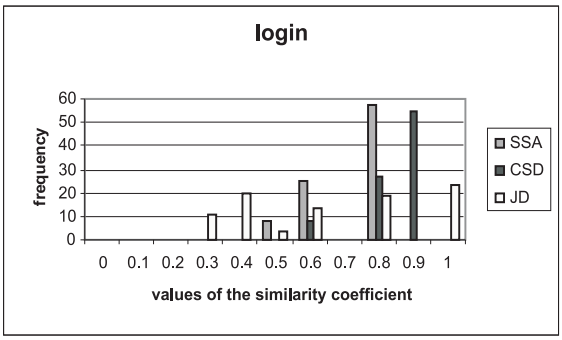

Fig. 8. The distance distributions of the *SSA*, *CSD* and *J* similarity coefficient for the process login

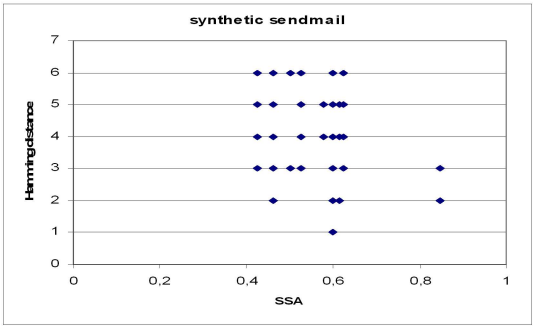

Fig. 9. The relation between the *SSA* similarity coefficient and the Hamming distance for the process synthetic sendmail

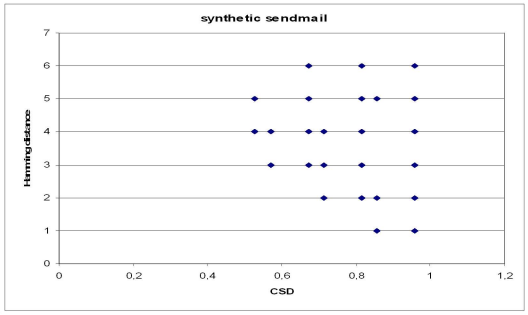

Fig. 10. The relation between the *CSD* similarity coefficient and the Hamming distance for the process synthetic sendmail

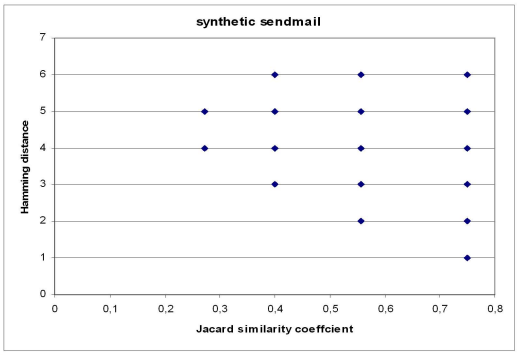

Fig. 11. The relation between the J similarity coefficient and the Hamming distance for the process synthetic sendmail

The detection rate represents the percentage of the false negatives regarding the true positives. The closer to 1 the detection rate is, the better the classification method is. All the presented values are between 0.700 and 0.996, which means that the proposed method can identify the intrusion patterns from the observed system call sequences with a very good detection rate. The lowest value is obtained for synthetic sendmail and *SSA* and *CSD* similarity coefficients.

The obtained values of *MCC* for all the processes belong to the interval (0.47; 0.87), which means balanced results, as a coefficient of +1 represents a perfect prediction. From Table 1 we can see that all *MCC* values obtained when the methodology applies the string distances are between 0.57 and 0.86. The highest values are obtained for the process named, the balanced results are obtained for the processes synthetic sendmail and inetd, and the lowest (but good enough) results are obtained for the process login. From Table 2 we can see that all *MCC* values obtained when the methodology applies the similarity measures are between 0.47 and 0.86. The *MCC* results obtained when the methodology applies the similarity measures are lower than the values obtained when the methodology applies the string distances. These results suggest that there is more significant correlation between the system behaviour profiles and the examined sequences of system activity, when the methodology applies the string distances.

The overall accuracy results are between 0.730 and 0.996, which suggest the number of false alarms is not significant, compared to the number of all predictions. From Table 1 we can see that all the overall accuracy values obtained when the methodology applies the string distances belong to the interval (0.9; 0.996). These results show the proposed methodology achieves an excellent level of diagnostic efficiency when the string metrics are applied, as the overall accuracy

Table 1. Detection rate, *MCC* and overall accuracy for the IDS based on the *WFD*, *JD* and *JWD*

synthetic sendmail	***Detection rate***	***MCC***	***Overall accuracy***
WFD	**0.9279**	0.7758	**0.9363**
JD	0.9228	**0.7836**	0.9331
***JWD* with factor** $p = 0.05$	0.8929	0.6721	0.9035
***JWD* with factor** $p = 0.1$	0.9151	0.7508	0.9253
named	***Detection rate***	***MCC***	***Overall accuracy***
WFD	**0.9823**	**0.8590**	**0.9832**
JD	0.9772	0.8286	0.9783
***JWD* with factor** $p = 0.05$	0.9726	0.8426	0.9745
***JWD* with factor** $p = 0.1$	0.9731	0.8446	0.9749
login	***Detection rate***	***MCC***	***Overall accuracy***
WFD	**0.9960**	0.6641	**0.9964**
JD	0.9951	0.5780	0.9950
***JWD* with factor** $p = 0.05$	0.9705	**0.7208**	0.9714
***JWD* with factor** $p = 0.1$	0.9701	0.7143	0.9711
inetd	***Detection rate***	***MCC***	***Overall accuracy***
WFD	**0.9910**	**0.7410**	**0.9912**
JD	0.9844	0.6818	0.9854
***JWD* with factor** $p = 0.05$	0.9861	0.7333	0.9863
***JWD* with factor** $p = 0.1$	0.9861	0.7330	0.9863

is the total probability that a system call pattern will be correctly classified by the proposed methodology. The values of the overall accuracy from Table 2 are between 0.733 and 0.978. These results are slightly lower than the results in Table 1, but still good enough, especially for the processes login and inetd, whose values belong to the interval (0.8; 0.978). These results suggest the application of the string metrics instead of the similarity coefficients achieves better results again.

In a *ROC* curve each *FAR* value can be plotted against its corresponding *HR* value for different cut-off points in order to create the diagrams. A test with perfect discrimination has a *ROC* plot that passes through the upper left corner. Therefore the closer the *ROC* plot is to the upper left corner, the higher the overall accuracy of the test [39]. Figures 12–14 contain the *ROC* curves for the processes synthetic sendmail, inetd and login, when the proposed methodology applies the similarity coefficients.

The IDS based on the *J* similarity coefficient achieves the 95% *HR* at the 35% *FAR* for the process login. The IDS based on the *SSA* similarity coefficient

Table 2. Detection rate, *MCC* and overall accuracy for the IDS based on the *SSA, CSD* and *J* similarity coefficients

synthetic sendmail	***Detection rate***	***MCC***	***Overall accuracy***
SSA	0.7000	0.4728	0.7334
CSD	0.7000	0.4728	**0.9330**
J	**0.8812**	**0.8732**	0.8447
named	***Detection rate***	***MCC***	***Overall accuracy***
SSA	**0.8841**	**0.6821**	0.8611
CSD	0.8438	0.5180	0.7941
J	0.8608	0.6265	**0.8681**
login	***Detection rate***	***MCC***	***Overall accuracy***
SSA	0.9647	0.4810	0.9546
CSD	0.9836	0.6120	**0.9780**
J	**0.9887**	**0.8624**	0.9733
inetd	***Detection rate***	***MCC***	***Overall accuracy***
SSA	0.9727	0.6463	0.9579
CSD	0.8386	0.5343	0.8065
J	**0.9728**	**0.6542**	**0.9631**

achieves the 77% *HR* at the 35% *FAR* for the process login. The IDS based on the *CSD* similarity coefficient achieves the 54% *HR* at the 35% *FAR* for the process login. Since the area under the *ROC* curves for the process login is from 0.9 to 1, this methodology represents an excellent result. The area under the *ROC* curves for the process synthetic sendmail is between 0.8 and 0.9, which means that this methodology gives good classification results. Since the area under the *ROC* curves for the process inetd in case of the *SSA* and *J* similarity coefficients is between 0.8 and 0.9, this methodology represents a good result, and in case of *CSD* similarity coefficients is between 0.7 and 0.8, which means a fair result.

The cost and its related hit rate are metrics of the performance of the applied methodology and the lower the cost, the better performance the proposed algorithm has. Table 3 presents the values of the cost for the examined processes. The best results are obtained when the proposed methodology applies the string distances *JD, WFD, JWD* with $p = 0.05$ and *JWD* with $p = 0.05$ respectively.

The results of the *SRCC* value are given in the Table 4.

The correlations between the results obtained by using different distances were all close to 1, making it evident that they are highly related.

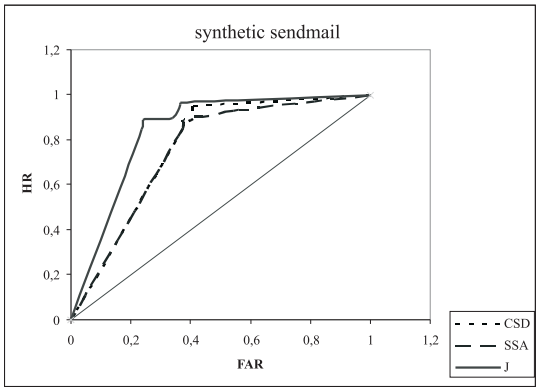

Fig. 12. The *ROC* curve for the process synthetic sendmail in the case of the *SSA*, *CSD* and *J* similarity coefficients

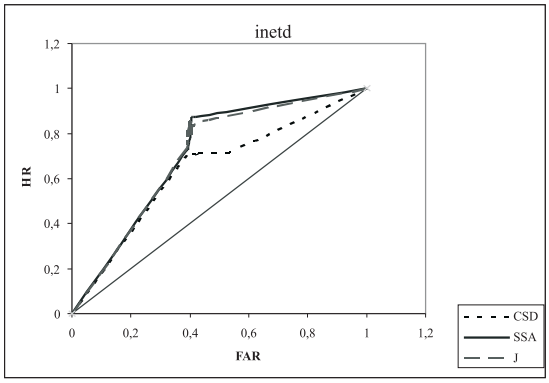

Fig. 13. The *ROC* curve for the process inetd in the case of the *SSA*, *CSD* and *J* similarity coefficients

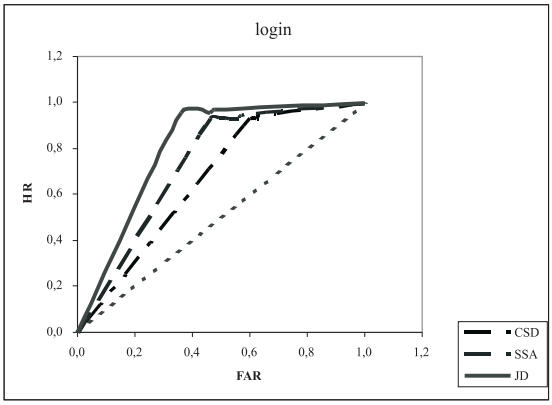

Fig. 14. The *ROC* curve for the process login in the case of the *SSA*, *CSD* and *J* similarity coefficients

Table 3. Cost for the processes synthetic sendmail, named, login and inetd

synthetic sendmail							
distance or coefficient	***WFD***	***JD***	***JWD p=0.05***	***JWD p=0.1***	***SSA***	***CSD***	***J***
Cost	0.3512	**0.3346**	0.4940	0.3840	0.9089	0.9089	0.9819
named							
distance or coefficient	***WFD***	***JD***	***JWD p=0.05***	***JWD p=0.1***	***SSA***	***CSD***	***J***
Cost	**0.2489**	0.6504	0.5750	0.5634	0.9737	0.9925	0.9400
login							
distance or coefficient	***WFD***	***JD***	***JWD p=0.05***	***JWD p=0.1***	***SSA***	***CSD***	***J***
Cost	0.6026	0.6643	**0.4647**	0.4740	1,0006	0.9950	0.9731
inetd							
distance or coefficient	***WFD***	***JD***	***JWD p=0.05***	***JWD p=0.1***	***SSA***	***CSD***	***J***
Cost	0.5000	0.5278	**0.4546**	0.6841	0.9952	0.9736	0.9923

VI. Discussions.

1. Comparison of the Applied Algorithms. Let's designate with A the IDS based on *WFD*, with B the IDS based on *JD*, with C the IDS based on *JWD* when p =0.05, with D the IDS based on *JWD* when p=0.1, with E the IDS based on *SSA* coefficient, with F the IDS based on *CSD* coefficient, and with G the IDS based on J coefficient.

The results, presented in Tables 1 and 2, show that all models achieve a very high level of detection rate results. The models which apply the string distances yield better results than the models which apply the similarity coefficients, as far as detection rate alone is concerned. The best results are achieved by model A for all the processes. We should mention that models B, C and D yield lower, but still very high detection rate levels, whose values are greater than 0.9 for all processes. Models E and G yield detection rate values greater than 0.9 for the processes login and inetd; and model F achieves detection rate values greater than 0.9 for the process login. Consequently, the testing results show that all models can identify the intrusion behaviours with very good detection rate.

The values of *MCC* presented in Tables 1 and 2 show balanced and reliable results for all models according to *MCC*, since all obtained values are between 0.47 and 0.86.

The SRCC values, presented in Table 4, reveal that there is a strong

Table 4. The values of *SRCC*

synthetic sendmail	***WFD***	***JD***	***JWD***	***SSA***	***CSD***	***J***
WFD	1					
JD	0.99	1				
JWD	0.98	0.99	1			
SSA	0.82	0.88	0.81	1		
CSD	0.84	0.87	0.82	0.98	1	
J	0.84	0.87	0.82	0.98	1	1
named	***WFD***	***JD***	***JWD***	***SSA***	***CSD***	***J***
WFD	1					
JD	0.98	1				
JWD	0.96	0.97	1			
SSA	0.85	0.89	0.83	1		
CSD	0.84	0.87	0.81	0.98	1	
J	0.83	0.86	0.82	0.99	1	1
login	***WFD***	***JD***	***JWD***	***SSA***	***CSD***	***J***
WFD	1					
JD	0.99	1				
JWD	0.98	0.98	1			
SSA	0.82	0.83	0.81	1		
CSD	0.87	0.87	0.79	0.97	1	
J	0.82	0.85	0.78	0.98	1	1
inetd	***WFD***	***JD***	***JWD***	***SSA***	***CSD***	***J***
WFD	1					
JD	0.97	1				
JWD	0.98	0.96	1			
SSA	0.88	0.88	0.81	1		
CSD	0.86	0.85	0.83	0.98	1	
J	0.86	0.85	0.81	0.96	1	1

relation between the results obtained by calculating the applied distances and similarity coefficients between the observed and normal sequences. As all SRCC values are between 0.8 and 1, the results indicate that the obtained distances and similarity coefficients for all examined processes are significantly associated with each other, since the value of 1 indicates that the two sets of data are identical.

Comparing the models' characteristics with respect to overall accuracy, we can conclude that model A yields the best results for all processes. Nevertheless, we should point out that all models A, B, C and D achieve excellent overall accuracy results, since the obtained values for all examined processes are greater than 0.9. In addition, model E yields overall accuracy results which are greater

than 0.9 for the processes login and inetd; model F yields overall accuracy results which are greater than 0.9 for the processes synthetic sendmail and login; and model G achieves overall accuracy results which are greater than 0.9 for the processes login and inetd.

The cost values, presented in Table 3, evaluate the performance of the proposed methodology depending on the applied distance or the similarity coefficient. Comparing the obtained values we could see that the most cost-effective technology is the one which applies the *WFD* for the process named. The lowest cost for *JD* is achieved for the process synthetic sendmail, and the lowest costs for the processes login and inetd are obtained when *JWD* with factor 0.05 is applied. Comparing the models A, B, C and D on the one hand, and E, F and G on the other hand, we observe the cost values for the first belong to the interval (0.24; 0.68), while the cost values for the second belong to the interval (0.9; 1). So we can conclude that applying the similarity coefficients is more cost-consuming, while applying the distances produces stable and more cost-effective results.

The results, obtained for models C and D and presented in Tables 2, 3 and 4, indicate that the values for all processes are very close. Consequently, the value of scaling factor p does not significantly influence the detection ability of the proposed methodology.

2. The algorithm complexity. The creation of the normal database, i.e. classification trees, requires $O(D)$ operations, where D is the number of the normal system calls, and $O(D*L)$ integer storage locations. The transition probabilities require a two-dimensional double array with $O(N^2)$ storage locations, where N is the number of the different system calls performed by the examined privilege process. These operations, however, are performed once during the initial system adjustment and configuration.

During the intrusion detection process we compare the normal activity sequences with those of the current activity. The methodology requires $O(\log N)$ time for each normal sequence extraction.

The Wagner Fischer algorithm requires $O(m*n)$ time and a *(m+1)*(n+1)* memory locations, where n and m are the lengths of the two strings. In our case the algorithm requires $O(L^2)$ time and a $(L+1)^2$ two-dimensional array for the storage of intermediate values for each comparison of sequences. The intrusion detection process using *JD* and *JWD* needs $O(L^2)$ time for the calculation of the distance. The similarity coefficients calculation requires $O(3L^2)$ operations for each comparison of sequences.

VII. Conclusion. Intrusion detection systems were first implemented in the early 90's. Since that time the field of research in intrusion detection

has focused on the ability to detect novel attacks and to minimize false alarms. The IDS are very successful tools as a second line of the system defence. Beside the attack detection, their purpose is to cause the attacker to spend a sufficient amount of resources in order to make the intrusion cost high enough. Finally, the actions of the intruder would be logged and analysed, which increases the potential risk for the attacker.

REFERENCES

[1] Anderberg M. R. Cluster Analysis for Applications. Academic Press, New York, 1973.

[2] Ariu D., G. Giacinto, R. Perdisci. Sensing Attacks in Computers Networks with Hidden Markov Models. *Machine Learning and Data Mining in Pattern Recognition*, ISBN: 978–3-540–73498–7, 2007, 449–463.

[3] Axelsson S. Intrusion Detection Systems: A Taxonomy and Survey. Technical Report No 99–15, Dept of Computer Engineering, Chalmers University of Technology, Sweden, March 2000.

[4] Bejtlich R. The Tao Of Network Security Monitoring: Beyond Intrusion Detection. Addison-Wesley Professional, 2004.

[5] Dice L. R. Measures of the amount of ecologic association between species. *Ecology*, **26** (1945), 297–302.

[6] Dasarathy B. V. Data Mining, Intrusion Detection, Information Assurance and Data Networks Security. In: Proceedings of the SPIE, **6973** (2008).

[7] Deng H. R., Y. H. Wang. An artificial-neural-network-based multiple classifiers intrusion detection system. In: Proceedings of the International Conference on Wavelet Analysis and Pattern Recognition (2007), ICWAPR'07, 683–686.

[8] Dhanalakshmi Y., I. R. Babu. Intrusion Detection Using Data Mining Along Fuzzy Logic and Genetic Algorithms. *International Journal of Computer Science and Network Security*, **8** (February 2008), No 2, 27–32.

[9] Feng H. H., O. M. Kolesnikov, P. Fogla, W. Lee, W. Gong. Anomaly detection using call stack information. In: Proceedings of the IEEE Symposium on Security and Privacy, Berkeley, CA (2003), 62–76.

[10] Feng L., X. Guan, S. Guo, Y. Gao, P. Liu. Predicting the intrusion intentions by observing system call sequences. *Computers & Security*, **23** (May 2004), Issue 3, 241–252.

[11] Ferri C., N. Lachinche, S. A. Macskassy, A. Rakotomamonjy (eds.) Second Workshop on ROC Analysis in ML, 2005.

[12] Forrest S., S. A. Hofmeyr, A. Somayaji, T. A. Longtaff. A Sense of Self for Unix Processes. In: Proceedings of IEEE Symposium on Security and Privacy (1996), IEEE Computer Society Press, Los Alamitors, CA, 120–128.

[13] Forrest S., S. A. Hofmeyr, A. Somayaji. Intrusion detection using sequences of system calls. *Journal of Computer Security*, **6** (1998), 151–180.

[14] Ghosh A.K., A. Schwartzbard, M. Schatz. Learning Program Behavior Profiles for Intrusion Detection. In: Proceedings of the 1st Workshop on Intrusion Detection and Network Monitoring (1999), 51–62.

[15] Gorodetsky V., O. Karsaeyv, V. Samoilov. Multi-agent technology for distributed data mining and classification. In: Proceedings of IEEE/WIC International Conference on Intelligent Agent Technology (2003), IAT 2003, 438–441.

[16] Haghighat A. T., M. Esmaeili, A. Saremi, V. R. Mousavi. Intrusion Detection via Fuzzy-Genetic Algorithm Combination with Evolutionary Algorithms. In: Proceedings of 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007), 587–591.

[17] Haslum K., A. Abraham , S. Knapskog. HiNFRA: Hierarchical Neuro-Fuzzy Learning for Online Risk Assessment. In: Proceedings of Second Asia International Conference on Modeling & Simulation AICMS 08, Kuala Lumpur (2008), 631–636.

[18] Hastie T., R. Tibshirani, J. H. Friedman. The Elements of Statistical Learning. Data Mining, Inference and Prediction, Springer, 2001.

[19] Jaccard P. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bull.Soc. Vaudoise Sci. Nat.,* **37** (1901), 547–579.

[20] Jaro M. A. Advances in record linking methodology as applied to the 1985 census of Tampa Florida. *Journal of the American Statistical Society*, 1989, 414–420.

[21] Kondrak, G., D. Marcu, K. Knight. Cognates Can Improve Statistical Translation Models. In: Proceedings of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003), 46–48.

[22] Khan L., M. Awad, B. Thuraisingham. A new intrusion detection system using support vector machines and hierarchical clustering. *The International Journal on Very Large Data Bases*, **16** (October 2007), Issue 4, 507–521.

[23] Khanna R., H. Liu. System approach to intrusion detection using hidden Markov model. In: Proceedings of the 2006 international conference on Wireless communications and mobile computing, Vancouver, Canada, 349–354.

[24] Kuang L., M. Zulkernine. An anomaly intrusion detection method using the CSI-KNN algorithm. In: Proceedings of the 2008 ACM symposium on Applied computing, Fortaleza, Ceara, Brazil, 921–926.

[25] Lee W. K., S. J. Stolfo. A data mining framework for building intrusion detection model. In: Proceedings of the IEEE Symposium on Security and Privacy.(Eds L. Gong, M. K. Reiter) Oakland, CA, IEEE Computer Society Press, 1999, 120–132,

[26] Leung K., C. Leckie. Unsupervised anomaly detection in network intrusion detection using clusters. In: Proceedings of the Twenty-eighth Australasian conference on Computer Science, **38** (2005), Newcastle, Australia, 333–342.

[27] Li X. B. A scalable decision tree system and its application in pattern recognition and intrusion detection, Decision Support Systems, **41** (November 2005), Issue 1, 112–130.

[28] Matthews B.W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim. Biophys. Acta*, 1975, 405, 442–451.

[29] Maxion R., T. Townsend. Masquerade detection using truncated command lines. In: Proceedings of international conference on dependable systems & networks, Washington DC, 2002, 219–228.

[30] Sekar R., M. Bendre, P. Dhurjati, D. Bullineni. A fast automaton-based method for detecting anomalous program behaviours. In: Proceedings of the IEEE Symposium on Security and Privacy S&P 2001, 144–155.

[31] Sorensen T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons. *K. Dan. Vidensk. Selsk. Biol. Skr.*, **5**, 1948, 1–34.

[32] Spearman C. The proof and measurement of association between two things. *Amer. J. Psychol.*, **15** (1904), 72–101.

[33] Taylor J. R. An Introduction to Error Analysis. The Study of Uncertainties in Physical Measurements, University Science Books, 1999, 128–129.

[34] Varghese S. M., K. P. Jacob. Anomaly Detection Using System Call Sequence Sets. *Journal of Software*, **2** (2007), Issue 6, 14–21.

[35] Venkatachalam V., S. Selvan. Intrusion Detection using an Improved Competitive Learning Lamstar Neural Network. *IJCSNS International Journal of Computer Science and Network Security*, **7** (February 2007), No 2, 255–263.

[36] Wagner R. A., M. J. Fischer. The string-to-string correction problem. *Journal of the Association for Computing Machinery*, **21** (1974), 168–173.

[37] Warrender C., S. Forrest, B. Pearlmutter. Detecting intrusions using system calls: Alternative data models. In: Proceedings of IEEE Symposium on Security and Privacy, IEEE Computer Society, 1999, 133–145.

[38] Winkler W. E. The state of record linkage and current research problems. Statistics of Income Division, Internal Revenue Service Publication R99/04, 1999.

[39] Zweig M.H., G. Campbell. Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. *Clinical Chemistry*, **39** (1993), No 4, 561–577.

Burgas Free University
Faculty for Computer Science and Engineering
62, San Stefano Str.
8001 Burgas, Bulgaria
e-mail: `vessi@bfu.bg`
e-mail: `enikolova@bfu.bg`

Received April 10, 2009
Finaly Accepted October 10, 2009