Autonomous decision on intrusion detection with trained BDI agents

Agustín Orfila [a] Javier Carbó [a] Arturo Ribagorda [a]

[a] *Computer Science Department, Universidad Carlos III de Madrid, Leganés 28911, Spain*

Abstract

In the context of computer security, the first step to respond to an intrusive incident is the detection of such activity in the monitored system. In recent years, research in intrusion detection has evolved to become a multi-discipline task that involves areas such as data mining, decision analysis, agent-based systems or cost-benefit analysis among others. We propose a multiagent IDS that considers decision analysis techniques in order to configure itself optimally according to the conditions faced. This IDS also provides a quantitative measure of the value of the response decision it can autonomously take. Results regarding the well known 1999 KDD dataset are shown.

Key words: Intrusion detection and response, multiagent system, decision analysis, knowledge management and reasoning

1 Introduction

Intrusion detection systems (IDS) are software or hardware systems that automate the process of monitoring the events occurring in a computer system or network, analysing them for signs of security problems [1]. A generic model of intrusion detection can be defined by a set of functions. These functions comprise raw data sourcing, event detection, analysis, data storage, and response. Data sources can be drawn from audit records and network traffic as well as from firewalls, switches and monitoring agents. The purpose of the event detection function is to provide relevant information for use in the analysis function. This may include eliminating unnecessary data and extracting relevant information from data sources. The data storage function is in charge of storing security-related information (such as audit logs, suspicious source sites, results of past analysis, and also profiles of known attacks and profiles of normal behaviour) making it available for analysis at a later time. Thus, the

analysis function processes data from both event detection and data storage functions. Furthermore, event detection and analysis functions can produce huge amounts of data that need the support of the storage function. Finally, the response function includes a decision module and provides countermeasure capabilities that are typically grouped into active and passive measures. Passive measures involve reporting IDS findings to humans, who are then expected to take actions based on these reports, while active measures involve some automated intervention in order to stop the progress of the intrusion.

Each IDS function can use methodologies and paradigms from different scientific disciplines. For instance, due to the nature of the event detection and analysis functions, data mining is an appropriate approach because it deals well with large amounts of information. Data mining is defined as the process of discovering patterns in data automatically. Specifically, the research in data mining applied to intrusion detection is related to the process of extracting the relevant security features from raw data as well as building effective and efficient machine learning algorithms to analyse the mined information. It is a very active research topic [2–6].

With respect to the response function, decision analysis [7,8] and cost-benefit modelling [9–12] are the most promising approaches. The former has been used to model the decision making process of taking actions against suspicious events and to evaluate IDS effectiveness. Nevertheless, this approach does not consider the cost involved in the response to a suspicious event. The latter has been used to calculate the cost of detecting and responding to an intrusion and to determine the trade-off between costs and benefits for a network IDS. The main handicap of the cost-benefit approach is the need to estimate every cost and risk involved in the process. In this paper, we propose a response function that simplifies the cost-benefit estimation while taking into account the most relevant costs for the decision making process.

For the knowledge management of the different IDS functions, the agent paradigm is a promising approach. Previous work in applying agents to intrusion detection was conducted at a number of research labs: the Autonomous Agents for Intrusion Detection (AAFID) effort at Purdue University [13], the Hummingbird project of the University of Idaho [14] and the Java Agents for Meta-Learning (JAM) project at Columbia University [3]. More recently, the main advances have been done on mobile distributed agents within the Mobile Agent Intrusion Detection Detection project at Iowa State University [15] and on lightweight agents at Queen's University in Canada [16].

Unfortunately, these projects do not accomplish deliberative reasoning of agents (their intelligence just relies upon learning) and do not deal directly with IDS effectiveness, what make it difficult to compare them with our system. Nevertheless, there are two previous agent-based IDS that satisfy one of both issues

and therefore, to the best of our knowledge, they are the only precedents of our work. One is the FAST system from the National Research Council of Canada [17]. FAST agents logically reason with the information provided by sensors at a higher level of abstraction in order to provide better analysis (due to information sharing). This is the case of deliberative agents whose intelligence relies upon an internal representation of the situation faced and the mental state in the form of beliefs, desires and intentions [18]. In their work, they do not show results about the system effectiveness. The other comes from the University of Hong-Kong [19], and shows results about effectiveness, although its agents make decisions with Fuzzy and Evolutionary Computation techniques instead of deliberative reasoning. We use the results of this study to measure the effectiveness of our design.

This paper proposes a deliberative multiagent system (MAS) which reasons in order to provide an effective IDS that configures itself optimally, according to the operating conditions. Moreover, it provides a quantitative estimation of the value of the response decision it can autonomously take. The remainder of this paper is organized as follows. Section 2 reviews the decision model analysis used by the response and analysis functions of the system we propose. Section 3 describes the MAS design and implementation issues. Section 4 presents the experimental setup. Section 5 shows results and discusses them and, finally, Section 6 summarizes the main conclusions.

2 Decision Model Analysis

Decision theory has been successfully applied in areas such as Psychology [20], Economy [21] or Meteorology [22]. In the computer security field, it has been used to face the intrusion detection task in order to provide a methodology to evaluate the effectiveness of different IDS under different operating conditions [7]. In this section, we review this methodology and extend it to include the response cost (the one incurred by taking action in order to avoid an intrusion). In addition, we introduce a useful metric (economic value) to measure IDS effectiveness.

The system to be protected can be in two possible states: an intrusive state (I) or a non intrusive state (NI). Similarly an IDS, depending on the analysis of data sources, can report an alarm (A) or not (NA). The conditional probabilities $P(A|I)$ (hit rate H) and $P(A|NI)$ (false alarm rate F) are the variables that define the detecting capabilities of the IDS. These probabilities are shown in Table 1. One of the main goals of any detection system is to achieve high values of H while keeping the F rate low. Nevertheless, as F increases so does H. On this basis, it is a desirable feature for an IDS to operate at different pairs (F, H). Each of these pairs is called an operating point. Thus, a detec-

Table 1
Conditional probabilities that an IDS detects the system state. F represents the false alarm rate and H the hit rate

	System state	
Detector's report	No intrusion (NI)	Intrusion (I)
No alarm (NA)	$1 - F$	$1 - H$
Alarm (A)	F	H

tor's ROC curve describes the relationship between its probability of detection (H) and its false alarm probability (F) for different operating points.

Lee [10] pointed out that a natural tendency in developing an intrusion detection system (IDS) is trying to maximize its *technical effectiveness* while neglecting the cost-benefit trade-off. An IDS needs to be cost-effective because it should cost no more than the expected level of loss from intrusions. This requires that an IDS considers the trade-off among cost factors, which at least should include the cost of damage caused by an intrusion, the cost of manual or automatic response to an intrusion, and the operational cost, which measures constraints on time and computing resources. Thus, damage cost characterizes the amount of damage to a target resource by a successful intrusion, whereas response cost is the cost of taking action in order to avoid an intrusion. Operational cost is the cost of the analysis and processing of the stream of events being monitored by the IDS. It is an important cost for evaluating the efficiency of an IDS but it is not relevant for evaluating the effectiveness.

Accordingly, we adopt a utility perspective in order to measure IDS effectiveness. Thus, the best IDS is the one that contributes most effectively to minimize expenses when defending a system. The expected cost of a detector on a certain operating point can be computed analysing the corresponding decision tree. Our tree is similar to the one proposed by Gaffney [7] but ours does not only consider the damage cost (L) but also the response cost (C). The decision tree relates the report of the IDS, the system state, the response of the decision module, the probabilities involved in the process and the consequence of all of the above (see Figure 1).

Decision or action nodes, which are displayed as squares, are under the control of the decision maker, who will choose which branch to follow. Conversely, the circles represent event nodes that are subject to uncertainty. A probability distribution represents the uncertainty about which branch will happen following an event node. Event node probabilities are defined as follows:

- p_1: is the probability that the detector reports no alarm.
- p_2: is the conditional probability of no intrusion given that the detector

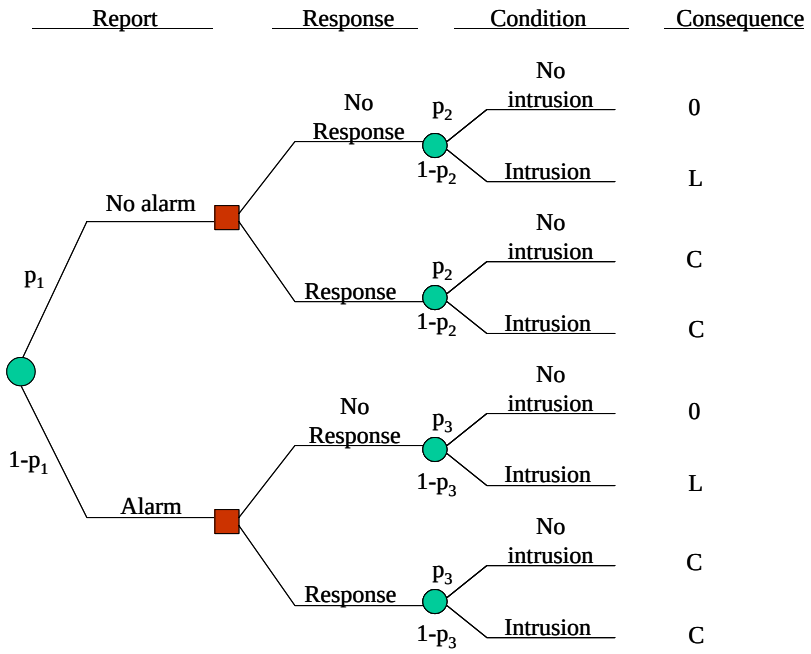

Fig. 1. Decision tree of the detector's expected cost that considers the response cost

Table 2
Expected cost of response decisions depending on the detector's report

	Response	
Detector's report	No	Yes
No alarm	$L(1-p_2) = \frac{L(1-H)p}{p_1}$	$Cp_2 + C(1-p_2) = C$
Alarm	$L(1-p_3) = \frac{LHp}{1-p_1}$	$Cp_3 + C(1-p_3) = C$

reports no alarm.

- p_3: is the conditional probability of no intrusion given that the detector reports an alarm.

An IDS can take some precautionary action depending on the likelihood of an intrusion occurring. Taking precautionary action incurs a cost C, irrespective of whether the intrusion occurs or not. However, if the intrusion occurs and no action has been taken, then a loss L is incurred. If there is no response and there is no intrusion, no cost is incurred. The decision maker (i.e. the network administrator or the decision module of the IDS if the response is automatic) will follow the strategy that minimizes the expected cost. In order to compute this expected cost it is necessary to calculate the expected cost conditional on the detector's report. The four possibilities are summarized in Table 2, where the prior probability of an intrusion happening is represented by p. The expected cost of each response is calculated by taking the sum of the products of the probabilities and costs for the node following each response.

Thus, if the report of the detector is known, the minimal expected cost can

be computed. If there is no alarm the expression for the expected cost under this condition is:

$$M_{NA} = \min\{L(1-p_2), C\} = \min\{\frac{L(1-H)p}{p_1}, C\} \tag{1}$$

Similarly, the expected cost given an alarm is:

$$M_A = \min\{L(1-p_3), C\} = \min\{\frac{LHp}{1-p_1}, C\} \tag{2}$$

Finally, the expected cost of operating at a given operating point, is the sum of the products of the probabilities of the detector's reports and the expected costs of operating conditioned by these reports. On this basis, the expected cost per unit loss (M) is:

$$\begin{aligned} M = &\min\{(1-H)p, \frac{C}{L}((1-F)(1-p) + (1-H)p)\} + \\ &+ \min\{Hp, \frac{C}{L}(F(1-p) + Hp)\} \end{aligned} \tag{3}$$

It is important to note that this formulation includes the possibility of taking actions against the report of the detector if these actions lead to a lower expected cost.

2.1 Metric of econmomic value

This subsection introduces a metric that measures the value of an IDS. In order to proceed, some concepts need to be defined first. The expected cost per unit loss of a perfect IDS (the one that achieves H=1 and F=0) is (from expression (3)):

$$M_{per} = \min\{p, \frac{C}{L}p\} = p\min\{1, \frac{C}{L}\} \tag{4}$$

In addition, an expression is needed for the expected cost when only information about the probability of intrusion is available (no IDS working). In this situation, the decision maker can adopt two alternative strategies: always protect, taking some precautionary action (incurring then in a cost C) or never

protect (incurring in losses pL). Consequently, the decision maker will respond if $C < pL$ and will not if $C > pL$. Then, the expected cost per unit loss is:

$$M_{prob} = \min\{p, \frac{C}{L}\} \tag{5}$$

Accordingly, the value of an IDS (V) is defined as the reduction it gives on the expected cost over the one that corresponds only to the knowledge of the probability of intrusion, normalized by the maximum possible reduction.

$$V = \frac{M_{prob} - M}{M_{prob} - M_{per}} \tag{6}$$

As a result, if an IDS is perfect at detecting intrusions its value is 1. Conversely, an IDS that does not improve a predictive system solely based on the probability of intrusion has a value less or equal to 0.

The metric of value is very useful because it includes all the relevant parameters involved in the evaluation of IDS effectiveness. A similar metric was proposed by the authors in [23] but it did not manage the possibility of a decision being made contrary to the detector's report as it is proposed in this publication.

3 The Multiagent System

3.1 Design of the Multiagent System (MAS)

Our system is composed of several cooperative agents that try to improve the overall IDS effectiveness through an autonomous adaptation. Agents play one of the following roles: sensor, evaluator or manager. We assume that several detection techniques are applied to the same source of information. Thus, each sensor agent applies a specific detection algorithm to infer a prediction about the intrusive nature of the analysed events. The predictions (often just a binary statement: intrusive/non-intrusive) are sent to evaluator agents that combine them to produce a final conclusion that is sent to the manager agent. Evaluator agents can apply different criteria to compute the conclusion. In this paper, two of them are considered:

- Threshold: the evaluator agent considers an event as an intrusion if the number of sensor agents that state the event as intrusive is greater than a

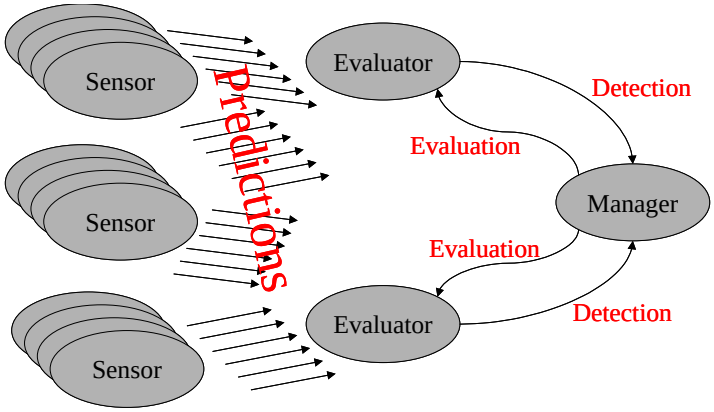

Fig. 2. Interactions between agents in the evaluation mode. Sensor agents make predictions about events and send them to evaluator agents that combine them to produce a final conclusion that is sent to the manager agent. Then, the manager informs the evaluator agents about their effectiveness

prefixed threshold.

- Weighted sum: the evaluator agent weights each sensor before the comparison with the prefixed threshold is done. The weights are updated after an event takes place according to the historical effectiveness of each sensor.

Finally, the manager agent may act in two different modes of behaviour: evaluation mode or operating mode. The former assumes the manager agent has the knowledge about the real nature of events, so it is able to inform the evaluator agents about their effectiveness. Evaluator agents will use this information to update the weights of sensor agents in order to improve future predictions. The latter consists of planning a response to intrusions, according to the beliefs previously acquired about the environment where events are taking place, and about the results provided in the evaluation mode. Figure 2 shows, in a general way, the interactions that take place between the agents of our system in the evaluation mode.

Let us now describe in detail the design of the multiagent system depending on the criteria used by the evaluator agent.

3.1.1 Evaluator agent applying thresholds

Parametric IDS are able to operate at different operating points. Axelsson [24] pointed out how important it is to tune IDS according to the environment faced. Sensor agents are based on different detection techniques that do not use parameters themselves. Nevertheless, the evaluator combines these techniques making the system parametric. The way the parameter is used to tune the IDS is now described. Assuming that sensor agents communicate binary statements (whether or not the events have an intrusive nature), the evaluator agent uses a probability threshold p_t to reach a final decision about the nature of

an event. If the percentage of sensor agents that consider this event as an intrusion is greater than this p_t, then the evaluator agent will consider it an intrusion. Accordingly, the hit and false alarm rates of the MAS considered as a single IDS become: $H = H(p_t)$ and $F = F(p_t) \quad \forall p_t \in [0, 1]$. Hence, the value of the resulting agent system also depends on such threshold as follows: $V = V(p_t) \quad \forall p_t \in [0, 1]$. Given a relationship between C and L, the optimum value will then be $V_{opt} = \max_{pt} V(p_t) \quad \forall p_t \in [0, 1]$. Thus, the manager agent may compute and show the results of the IDS effectiveness in the evaluation mode for different p_t. In the case of sensors that classify events according to specific types of intrusion, the evaluator agent would observe which of these types is the one detected by the majority of the sensor agents. It would then reach a final conclusion about the nature of the event according to the threshold, similarly to the process previously described.

3.1.2 Evaluator agent applying dynamic weights

In this setup, we assume that sensor agents that were more successful in the past, are also going to be more successful with future events. This assumption is the motivation of the adaptive computing of sensor agent weights. These weights are computed using the economic value of the optimal operating point. For the model presented in Section 2, this point corresponds to $\frac{C}{L} = p$. Thus, the adaptive process depends on the optimal operating point of each sensor, the probability of intrusion, and the damage and response costs corresponding to this operating point. As events are taking place, the economic value of each sensor agent will change and, therefore, the influence of each sensor in the computing of the evaluator's final conclusion.

If sensor agents distinguish between different types of intrusion, the evaluator agent would weight each sensor agent according to the level of success for each type of intrusion. Hence, the evaluator agent would compute the average sum of the sensor agent weights for each type. The type with the greatest value would be the candidate to be predicted. If the weighted sum for this type of intrusion overcomes the corresponding threshold, the evaluator agent would state that there is an intrusion.

We cannot assume that the manager agent has *a posteriori* information about the intrusive nature of every event (otherwise, the complete IDS would not make any sense). Therefore, the adaptation process should be done off-line under a training scenario where the manager agent knows the nature of the processed events. Once the MAS has been trained, weights are fixed and the IDS can be tested under realistic conditions. Our experiments test the effectiveness of this approach.

3.2 Reasoning model of agents

Research on agents can be split into two main trends: one is related to agent reaction facing external stimulus and the other is focused on agents with symbolic internal models. The intelligence produced by these internal models is based on a deliberation about the state of the outside world (and its past evolution), and the events that may take place in the future. Since in our domain evaluator agents need some deliberation about past observations to improve their future behaviour, the use of a memory of past changes in the outside world, and a certain level of planning becomes appropriate. Therefore, the agents of our IDS multiagent system should be intelligent or, in other words, they should make use of a symbolic internal model.

To build agents with such deliberation ability, several architectures, inspired from different disciplines such as psychology, philosophy and biology, can be applied. Most of these are based on theories for describing the behaviour of individuals. Among them, we have chosen the BDI model [18] to implement the deliberation of our agents since it is, by far, the most popular way of implementing deliberative agents due to its simplicity and psychological background. In order to act rationally, the BDI model represents the situation faced internally and the mental state in the form of beliefs, desires and intentions.

Let us now outline the sequence of intentions forming the plans of sensor, evaluator and manager agents.

- The plan that produces the corresponding predictions of sensor agents is composed only of a single chain of three intentions:
  - Communicative Intention *Wait_event*: Wait until a new event is received from the evaluator agent.
  - Internal Intention *Predicts*: Compute the prediction about the intrusive nature of the event.
  - Communicative Intention *Send_prediction*: Send the corresponding prediction to the evaluator agent.
- The plan that combines the conclusions from sensor agents into a final prediction to be sent to the manager agent:
  - Communicative Intention *Wait_event*: Wait until a new event is received from the manager agent.
  - Communicative Intention *Send_request_prediction*: For each sensor agent that belongs to the system: ask for a prediction and wait for the corresponding answer.
  - Internal Intention *Combine_predictions*: Compute the final detection for different thresholds (p_t), applying the corresponding weights to the conclusions received from the sensor agents.
  - Communicative Intention *Send_detection*: Send the final detection to the

manager agent.
  - Communication Intention *Wait_HVF*: Wait for the H, F and V values for each agent from the manager agent.
  - Internal Intention *Update_weights*: Update the weights of sensor agents according to their H, F, and V values.
- The plan followed by the manager agent consists of the next cycle:
  - Communicative Intention *Send_event*: Send a new event to the evaluator agent.
  - Communicative Intention *Wait_detection*: Wait for the corresponding message from the evaluator agent which includes a final conclusion about the intrusive nature of the event.
  - Internal Intention *Apply_detection*: Compute H, F and V for each agent, including the sensor agents and the evaluator agent.
  - Communicative Intention *Send_HVF*: Send H, V and F to the evaluator agent.

When our agents become trained, they will act in operating mode, where the reasoning of the manager and the evaluator agent is different. It is now orientated to the selection of the best automatic response to incidents while evaluator agents do not change the weights along the execution. Therefore, the role of the manager agent is focused on the right tuning of the optimal operating point with the current beliefs about the characteristics of the environment and with data about the economic value of the system, previously obtained during the evaluation mode.

Thus, the plan followed by the evaluator agent in operating mode will be:

- Communicative Intention *Wait_tuning_information*: Wait until the corresponding tuning information arrives from the manager agent.
- Communicative Intention *Wait_event*: Wait until a new event is received from the manager agent.
- Communicative Intention *Send_request_prediction*: For each sensor agent that belongs to the system: ask for a prediction, wait for the corresponding answer.
- Internal Intention *Combine_predictions*: Compute the final detection with the received p_t threshold, applying fixed weights to the conclusions received from the sensor agents.
- Communicative Intention *Send_detection*: Send the final detection to the manager agent.

The last two intentions of the plan that was applied in the evaluation mode are now ignored. The plan followed by the manager agent in operating mode will be:

- Communicative Intention *Send_tuning_information*: According to its beliefs,

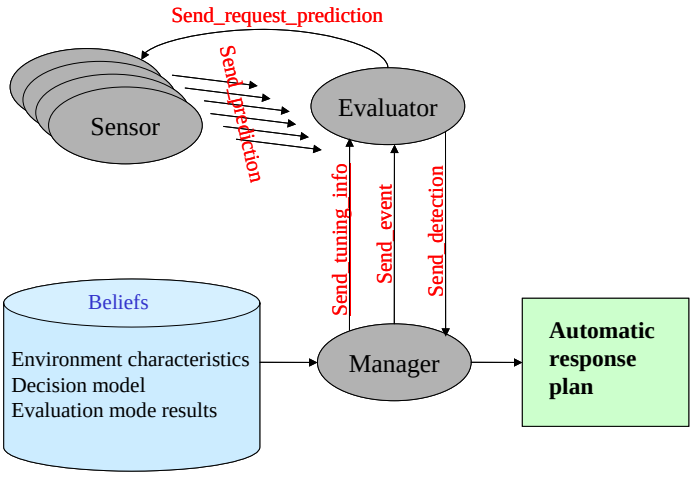

Fig. 3. Communicative intentions of agents in the operating mode. The manager agent sends the tunning information to the evaluator agent according to its beliefs. When a new event arrives, the manager sends it to the evaluator and waits for its opinion. Finally, the autonomous response plan taken by the manager depends on this opinion and its beliefs about the environment, the decision model and the results of the evaluation mode

send the optimum p_t threshold to the evaluator agent.

- Communicative Intention *Send_event*: Send a new event to the evaluator agent.
- Communicative Intention *Wait_detection*: Wait for the corresponding message from the evaluator agent which includes a final conclusion about the intrusive nature of the event.
- Internal Intention *Create_plan*: analyse both the internal beliefs about the characteristics of the environment and the economic value information available about the system. Apply then the decision model in order to produce a plan automatically to react or not to the incident.

Figure 3 shows graphically the interactions that take place in the operating mode.

3.3 Implementation of the multiagent system

The foundation for most implemented BDI systems is the abstract interpreter proposed by Rao and Georgeff (see Algorithm 1) [25]. Although many adhoc implementations of this interpreter have been applied to several domains, the recent release of JADEX [26] is obtaining increasing acceptance. JADEX is an extension of JADE [27] which facilitates FIPA communications between agents, and is widely used to implement intelligent and software agents. But JADEX also provides a BDI interpreter for the construction of agents. The beliefs, desires and intentions of JADEX agents are defined easily with XML and

Table 3
FIPA-ACL messages exchanged between agents in the proposed system

sender	*receiver*	Communicative act	*Content*
Manager	Evaluator	*request*	Detection
Evaluator	Sensor	*request*	Prediction
Sensor	Evaluator	*inform*	Prediction
Evaluator	Manager	*inform*	Detection
Evaluator	Manager	*inform*	Prediction
Manager	Evaluator	*inform*	H,F,V

Java enabling researchers to quickly exploit the potential of the BDI model. It is a promising technology that may soon become an unofficial standard on which to build deliberative agents.

Thus, the final implementation of the multiagent system includes 5 plans: sensor, evaluator-evaluation-mode, manager-evaluation-mode, evaluator-operation-mode and manager-operation-mode. The corresponding intentions of these plans: 9 communicative intentions, and 5 internal intentions, were implemented as JAVA methods integrated into JADEX infrastructure. Finally, firing conditions of plans, and the definition of beliefs is defined in a single XML file.

The 6 messages exchanged between agents in our system adopt the performatives shown in Table 3.

4 Experimental Setup

Any experiment related to IDS effectiveness faces similar problems. It is difficult to use real data for evaluation purposes because of privacy concerns. An alternative possibility is to use artificial data but the similarity of artificial events with real ones remains an open question. Therefore, adhoc methodologies that use proprietary data remain prevalent, making it very difficult to evaluate the significance of the different proposals [28]. Another problem is to choose a unit of analysis. Each IDS analyses the data sources at a certain level. Thus, the analysis process is done at different network layers and with different logging depth depending on the IDS. As a consequence, comparison is not easy. An exhaustive review of the inherent difficulties for IDS evaluation can be found in [29]. Although the handicaps related to artificial data and the difficulties associated to building a universal benchmark, there is a need of public datasets. In fact, there is a frequently used one. It was created in

1998 and 1999 in MIT Lincoln Laboratories in order to test different IDS [30]. A military network was simulated. Although this artificial dataset has been criticized [31,32], it has been and is still used by the scientific community. It is also important to note that security threats have changed since 1999 although the dataset is still valuable as a public reference.

In order to model the sensor agents of our system, different detection techniques were used on the same dataset. As previously mentioned, it was important that these detection techniques shared a common unit of analysis. For this reason, we chose for the experiments the well known 1999 KDD dataset [1] [33] that derives from MIT/LL 98 evaluation. The KDD dataset is the most frequently used dataset to test machine learning algorithms in the intrusion detection domain (e.g. [34–36]). Training and testing datasets were created at Columbia University. The KDD dataset was first employed for a machine learning competition in order to test different classifiers over the intrusion detection domain. A complete description of the data mining process can be found in [37] and is currently available at California University website [2]. Let us now review the dataset briefly (a general description can be found in [33]). Each connection record defines a TCP session and is described by 41 attributes (38 numeric and 3 nominal), and the corresponding class which indicates if the record represents normal or hostile activity. The number of normal and attack examples are summarized in Table 4. As can be seen, the percentage of attacks is extraordinary high both in training and test datasets. This situation is not expected in a real environment because the probability of intrusion is usually very low. However, we have shown in Section 2 the importance of the probability of intrusion (p) when evaluating IDS effectiveness. Consequently, the experiments we carried out were done over original and filtered data. The former allowed the comparison with previous research whereas the latter focused on a more realistic situation. The filtering process consisted of getting rid of the most common attack types both in training and test datasets in order to get an attack rate under 5% (this filtering rate has also been chosen by [35]). It is important to comment that the detection process becomes a harder task after filtering. The resulting number of examples after filtering the original dataset is summarized in Table 5.

The experimental setup is the same regardless of whether the data is filtered or not. On the one hand, 10 sensor models were built from the training dataset distinguishing normal from intrusive events. These correspond to 10 different machine learning algorithms. Two are based on decision trees (*ADTree, J48*), five on rules (*ConjuctiveRule, DecisionStump, DecisionTable, OneR, PART*), one on bayesian learning (*NaïveBayes*) and two on simpler techniques (*Hyper-*

[1] In fact the reduced version of the dataset (10% of the complete one)
[2] `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`

Table 4
Number of attack and normal instances in the original KDD training and test datasets. It is important to note that the percentage of attacks in both datasets is over 80%

	Training	Test
Normal instances	97277	60593
Attack instances	396743	250436
Total	494020	311029
% of normal instances	19.69	19.48
% of attack instances	80.31	80.52

Table 5
Number of attack and normal instances in the filtered training and test datasets. After the filtering process the percentage of attacks remains under 5% in both datasets

	Training	Test
Normal instances	97277	60593
Attack instances	4887	2650
Total	102164	63243
% of normal instances	95.22	95.81
% of attack instances	4.78	4.19

Pipes, VFI)[3]. It is important to note that these models were not customized in order to optimize their effectiveness. On the other hand, the test dataset was used to test the multiagent system in the evaluation mode. Four different experiments were done. First, the sensor models were tested alone and compared by means of the metric of value defined in Section 2. Second, the MAS in the configuration in which the evaluator agent did not update the weights for the sensors (threshold criteria) was considered. Third, the MAS with the weighted sum criteria was analysed over the whole test dataset. Finally, the fourth experiment took into account that usually there is only partial forensic information available in order to adapt the decisions of the evaluator agent. Therefore, the weight adaptation was made just over 10% of the test dataset.

[3] The names correspond to the implementation name in WEKA software [38]

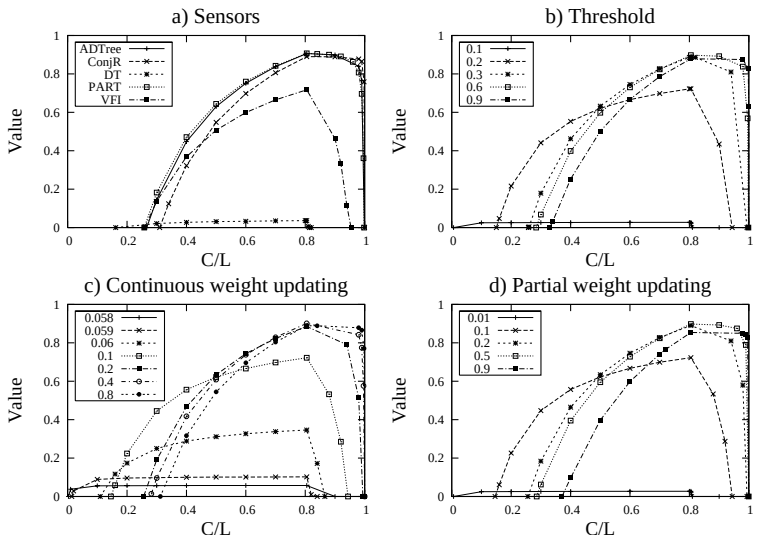

Fig. 4. Value curves of the four proposed configurations computed over the original test KDD dataset. a) represents the results of the more significant sensor agents trained over the original training dataset. b), c) and d) show the results of the MAS in the three different configurations for those p_t that contribute to form the envelope. Specifically, b) corresponds to the MAS in the pure threshold configuration, c) to the MAS with a continuous update of the sensors weights over the original test dataset and d) to the MAS with a partial weight update. The envelope of the MAS value curves in each configuration represents the MAS effectiveness

5 Results

First, the results obtained for the original dataset in the evaluation mode are exposed. Figure 4 shows the value curves that correspond to the four experiments previously described. Value curves represent the value of each system versus different $\frac{C}{L}$ relationships. For the three MAS settings, results are shown for different tuning of the parameter p_t. Thus, if $\frac{C}{L} > 0.5$, the results for the best sensor ($PART$) are very similar to the MAS in any of its three configurations (Figure 6 b), c) and d)). Nevertheless, for $\frac{C}{L} < 0.5$ our multiagent approach clearly outperforms any of the sensors. Furthermore, in this case, the continuous weight updating configuration overcomes both the threshold and partial weight updating configuration. For instance, for $\frac{C}{L} = 0.2$ all the sensors are worthless, but the MAS has a value that is about 20% of a perfect IDS.

Let us now compare the results of our system with previous research. The comparison is done against two systems that use the original KDD test dataset for their experimental work. One is the winner of the KDD cup learning contest [39] and the other is the agent-based system MOGF-IDS [19]. Figure 5 shows the value curves of our multiagent approach (with partial weight updating) versus these systems. As can be seen from the envelope of the MAS, our

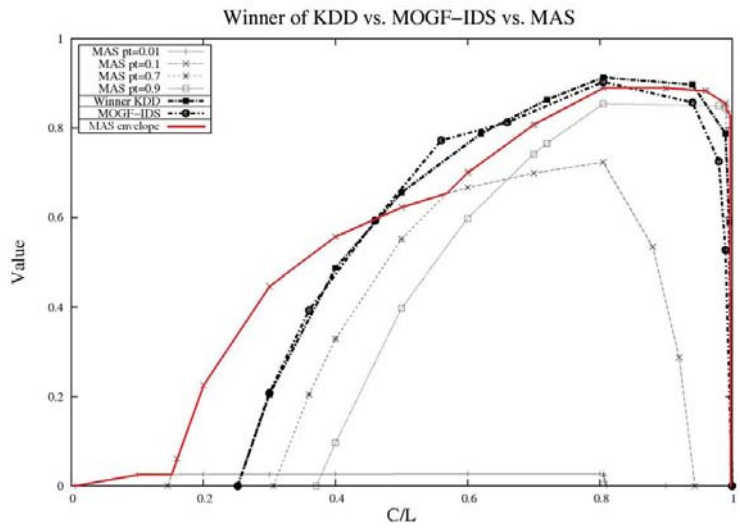

Fig. 5. Comparison of the KDD winner, the MOGF-IDS and the MAS with partial weight updating. The MAS system outperforms the KDD winner and the MOGF-IDS for low and high $\frac{C}{L}$ relationships because, in such cases, the envelope of the MAS value curves covers the value curve of the other two systems

system is the most effective for low $\frac{C}{L}$ relationships ($\frac{C}{L} < 0.45$). Contrary, for intermediate $\frac{C}{L}$, either the winner of the KDD or MOGF-IDS outperforms the MAS. Finally, for $\frac{C}{L} > 0.95$ our system is again the best one. It is important to note that the choice of the sensor agents was not optimized. For instance, if MOGF-IDS had been used as a sensor agent (unfortunately the code is not publicly available), it is reasonable to think that the MAS effectiveness would have improved. From this point of view, the results of our system are even better.

Second, the results in the evaluation mode obtained filtering the dataset are exposed. Although many studies filter the dataset, it is not easy to compare

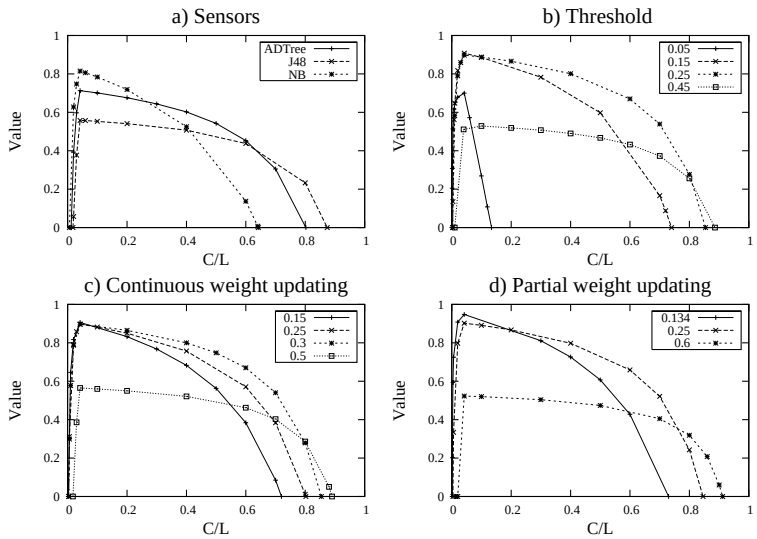

Fig. 6. Value curves of the four proposed configurations computed over the filtered test KDD dataset. a) represents the results of the best three sensor agents trained over the filtered training dataset. b), c) and d) show the results of the MAS in the three different configurations for those p_t that contribute to form the envelope. Specifically, b) corresponds to the MAS in the pure threshold configuration, c) to the MAS with a continuous update of the sensors weights over the filtered test dataset and d) to the MAS with a partial weight update. The envelope of the MAS value curves in each configuration represents the MAS effectiveness

results because the resultant filtered datasets are not usually available [4]. As a consequence, results are presented for the four experiments referred in Section 4. Figure 6 shows the corresponding value curves.

Threshold MAS clearly outperforms any of the sensor models alone. This statement derives from the fact that the envelope of the threshold MAS curves includes the envelope of the individual model composition. In other words, there is no cost relationship where a single sensor outperforms the MAS. The maximum value of the threshold MAS is 9.3% higher than the maximum of the best sensor (*Naïve Bayes*) and the range with positive value is also bigger. In addition, for low $\frac{C}{L}$ relationships ($0.00012 < \frac{C}{L} < 0.018$), none of the sensors separately gets possitive value in contrast to the MAS. There are many scenarios where the response cost is much lower than the cost of suffering an intrusion. Hence, this is an important advantage of the threshold MAS over the individual models.

Both the MAS with continuous weight updating and partial weight updating obtain similar results. The envelope of curves in Figure 6d) covers the envelope of curves in Figure 6b). Therefore, the adaptation process obtains good results.

[4] The original and filtered datasets we have used are available at `http://www.lab.inf.uc3m.es/~adiaz/ids/index.htm`

Thus, the increase in range with positive value is 5% and the maximum value is 3.9% higher. The advantage of a greater range is the IDS is valuable under more different operating conditions. For instance, a denial of service attack often produces more damage to an e-commerce site than to a personal website. The response cost for stopping the attack is similar in both cases (perhaps the cost of filtering traffic from a certain subnet), but the damage cost of not giving service for hours is clearly higher for the e-comerce site. Therefore, $\frac{C}{L}$ relationship depends on the specific scenario and so does IDS effectiveness. It is a desirable feature for an IDS to be effective in both scenarios.

To summarize, the multiagent system is more effective than any of the single sensor it is composed of. Furthermore, better results are achieved if the evaluator agent decision is taken updating the influence of each sensor agent on their past success.

Third, let us now explain how the decision response is taken in the operating mode. The manager beliefs are the value curves previously described (obtained in the evaluating mode), the decision model of Section 2 and an estimation of $\frac{C}{L}$ relationship for the environment faced. For instance, if the manager beliefs are the curves in Figure 6d) and the response cost is a hundred times smaller than damage cost ($\frac{C}{L} = 0.01$), then the tuning information sent to the evaluator agent will be $p_t = 0.134$ (because this p_t provides the greatest possible value). When the manager asks the evaluator if it considers an event as intrusive, the answer will be made according to this threshold. The decision (to respond or not) will be taken according to Table 2 (updating H and F considering the current event). According to Figure 6d), this decision will have a value that is 82.9% of the one expected from a perfect IDS.

Possible responses include to notify the alarm to a human expert, to make the system collect additional information, to change the environment (e.g. reconfigure the routers and firewalls) or even to disconnect the system from the network to protect it. Each measure will have a different response cost.

6 Conclusion

This paper has proposed a multiagent system that covers the analysis and response functions of an IDS. The deliberative nature of the implemented agents explains the reasoning of the IDS both in the processes of detection and response. The adaptive process based on the past success of sensors has been proved to be a good strategy. In addition, although the use of agents imposes an operational cost overhead, we have shown that the effectiveness of decisions taken by the MAS during an evaluation stage provides knowledge to configure the IDS optimally to face different operating conditions. Thus, the presented

IDS is able to autonomously decide if it is appropriate to take actions against a suspicious event or not, while assessing the value of the decision. This decision is taken reasoning about the evaluation stage results, the probability of intrusion, the costs of the environment faced, and the decision model adopted.

For both the analysis and response functions, a decision model that considers cost-benefit analysis has been proposed. The main problem of cost-benefit models is the need to estimate too many parameters. Our approach only needs to estimate the relationship between the damage and response cost to evaluate IDS effectiveness.

This paper has also introduced a metric of economic value that allows to know how far an IDS is from the perfect one, in what conditions an IDS is worthless and how to compare IDS effectiveness. Results regarding this metric have been presented for the well-known KDD dataset. The effectiveness of our system is comparable to state of the art approaches, overcoming them or not depending on the operating conditions.

The sensor agents of our system were trained with different supervised machine learning algorithms in order to classify events just as normal or intrusive. Future work will consider the different nature of attack events. Thus, the sensor agents will be trained to distinguish between different types of intrusion. The analysis and response processes will surely become more complex but hopefully more accurate.

References

[1] R. Bace, P. Mell, NIST special publication on intrusion detection system, Tech. rep., NIST (National Institute of Standards and Technology), Special Publication 800-31 (August 2001).

[2] W. Lee, A data mining framework for building intrusion detection models, in: Proceedings of the 1999 IEEE Symposium on Security and Privacy, Berkeley, California, USA, 1999, pp. 120–132.

[3] S. J. Stolfo, W. Lee, P. K. Chan, W. Fan, E. Eskin, Data mining-based intrusion detectors: an overview of the Columbia IDS project, SIGMOD Rec. 30 (4) (2001) 5–14.

[4] M. A. Maloof, Machine Learning and Data Mining for Computer Security: Methods and Applications (Advanced Information and Knowledge Processing), Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[5] G. Simon, H. Xiong, E. Eilertson, V. Kumar, Scan detection: A data mining approach, in: Proceedings of the Sixth SIAM International Conference on Data Mining, 2006, pp. 118–129.

[6] T. Özyer, R. Alhajj, K. Barker, Intrusion detection by integrating boosting genetic fuzzy classifier and data mining criteria for rule pre-screening, Journal of Network and Computer Applications 30 (1) (2007) 99–113.

[7] J. E. Gaffney, J. W. Ulvila, Evaluation of intrusion detectors: A decision theory approach, in: Proceedings of the IEEE Symposium on Security and Privacy, SP '01, IEEE Computer Society, Washington, DC, USA, 2001, pp. 50–.

[8] J. W. Ulvila, J. E. Gaffney, A decision analysis method for evaluating computer intrusion detection systems, Decision Analysis 1 (1) (2004) 35–50.

[9] H. Wei, D. Frinke, O. Carter, C. Ritter, Cost-benefit analysis for network intrusion detection systems, in: CSI 28th Annual Computer Security Conference, Washington, D.C., USA, 2001.

[10] W. Lee, W. Fan, M. Miller, S. J. Stolfo, E. Zadok, Toward cost-sensitive modeling for intrusion detection and response, Journal of Computer Security 10 (1-2) (2002) 5–22.

[11] C. Iheagwara, A. Blyth, M. Singhal, Cost effective management frameworks for intrusion detection systems, Journal of Computer Security 12 (5) (2004) 777–798.

[12] N. Stakhanova, S. Basu, J. Wong, A cost-sensitive model for preemptive intrusion response systems, in: Proceedings of the 21st International Conference on Advanced Networking and Applications, AINA '07, IEEE Computer Society, Washington, DC, USA, 2007, pp. 428–435.

[13] J. Balasubramaniyan, J. O. Garcia-Fernandez, D. Isacoff, E. H. Spafford, D. Zamboni, An architecture for intrusion detection using autonomous agents, Tech. Rep. Coast TR 98-05, The COAST Project, Department of Computer Sciences, Purdue University, West Lafayette, USA (1998).

[14] D. Frincke, D. Tobin, J. McConnell, J. Marconi, D. Polla, A framework for cooperative intrusion detection, in: Proceedings of the 21st National Information Systems Security Conference, 1998, pp. 361–373.

[15] Y. Wang, S. R. Behera, J. Wong, G. Helmer, V. Honavar, L. Miller, R. Lutz, M. Slagell, Towards the automatic generation of mobile agents for distributed intrusion detection system, Journal of Systems and Software 79 (1) (2006) 1–14.

[16] M. Shyu, T. Quirino, Z. Xie, S. Chen, L. Chang, Network intrusion detection through adaptive sub-eigenspace modeling in multiagent systems, ACM Transactions on Autonomous Adaptive Systems 2 (3) (2007) 9.

[17] M. Shajari, A. A. Ghorbani, Application of belief-desire-intention agents in intrusion detection and response, in: Proceedings of the Second Annual Conference on Privacy, Security and Trust, PST'04, University of New Brunswick Fredericton, Canada, 2004, pp. 181–191.

[18] M. E. Bratman, Intentions, Plans, and Practical Reason, Harvard University Press, Cambridge, Massachusetts, USA, 1987.

[19] C. Tsang, S. Kwong, H. Wang, Anomaly intrusion detection using multi-objective genetic fuzzy system and agent-based evolutionary computation framework, in: Proceedings of the Fifth IEEE International Conference on Data Mining, Houston, USA, 2005.

[20] J. A. Swets, R. Dawes, J. Monahan, Psychological science can improve diagnostic decisions, Psychological Science in the Public Interest 1 (1) (2000) 1–26.

[21] A. Sen, Choice functions and revealed preferences, Review of Economic Studies 38 (1971) 307–317.

[22] R. W. Katz, A. H. Murphy, Economic value of weather and climate forecasts, Cambridge University Press, UK, 1997.

[23] A. Orfila, J. Carbó, A. Ribagorda, Fuzzy logic on decision model for ids, in: Proceedings of the Twelveth IEEE International Conference on Fuzzy Systems, FUZZ-IEEE '03, Vol. 2, St. Louis, Missouri, USA, 2003, pp. 1237–1242.

[24] S. Axelsson, The base-rate fallacy and the difficulty of intrusion detection, ACM Transactions on Information and System Security, TISSEC 3 (3) (2000) 186–205.

[25] A. S. Rao, M. P. Georgeff, An abstract architecture for rational agents, in: B. Nebel, C. Rich, W. Swartout (Eds.), Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning, KR '92, Morgan Kaufmann publishers Inc., Cambridge, Massachusetts, USA, 1992, pp. 439–449.

[26] A. Pokahr, L. Braubach, W. Lamersdorf, Jadex: Implementing a BDI-infrastructure for JADE agents, EXP - in search of innovation (Special Issue on JADE) 3 (3) (2003) 76–85.

[27] F. Bellifemine, A. Poggi, G. Rimassa, JADE - a FIPA-compliant agent framework, in: Proceedings of the Practical Applications of Intelligent Agents and MultiAgents, PAAM '99, London, UK, 1999, pp. 97–108.

[28] N. Athanasiades, R. Abler, J. G. Levine, H. L. Owen, G. F. Riley, Intrusion detection testing and benchmarking methodologies, in: Proceedings of the International Information Assurance Workshop, IWIA '03, Maryland, USA, 2003, pp. 63–72.

[29] P. Mell, V. Hu, R. Lippman, J. Haines, M. Zissman, An overview of issues in testing intrusion detection, Tech. rep., National Institute of Standards and Technologies. Internal report 7007 (Jun 2003).
URL `http://csrc.nist.gov/publications/nistir/nistir-7007.pdf`

[30] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, K. Das, The 1999 DARPA off-line intrusion detection evaluation, Computer Networks 34 (4) (2000) 579–595.

[31] J. McHugh, Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory, ACM Transactions on Information and System Security 3 (4) (2000) 262–294.

[32] M. V. Mahoney, P. K. Chan, An analysis of the 1999 darpa/lincoln laboratory evaluation data for network anomaly detection., in: G. Vigna, E. Jonsson, C. Krügel (Eds.), Proceedings of the Sixth International Workshop on Recent Advances in Intrusion Detection, Vol. 2820 of Lecture Notes in Computer Science, Springer, Pittsburgh, USA, 2003, pp. 220–237.

[33] C. Elkan, Results of the KDD'99 classifier learning contest, http://www-cse.ucsd.edu/users/elkan/clresults.html (September 1999).

[34] G. Giacinto, F. Roli, L. Didaci, Fusion of multiple classifiers for intrusion detection in computer networks, Pattern Recognition Letters 24 (12) (2003) 1795–1803.

[35] P. Laskov, P. Düssel, C. Schäfer, K. Rieck, Learning intrusion detection: supervised or unsupervised, in: Proceedings of the Thirteenth International Conference on Image Analysis and Processing, ICIAP 2005, Cagliari, Italy, 2005.

[36] T. Khoshgoftaar, K. Gao, H. Lin, Indirect classification approaches: a comparative study in network intrusion detection, International Journal of Computer Applications in Technology 27 (4) (2006) 232–245.

[37] W. Lee, S. J. Stolfo, A framework for constructing features and models for intrusion detection systems, ACM Transactions on Information and System Security 3 (4) (2000) 227–261.

[38] I. H. Witten, E. Frank, Data mining: practical machine learning tools and techniques, Morgan Kaufmann Publishers Inc., San Francisco, California, USA, 2005.

[39] C. Elkan, Results of the kdd'99 classifier learning, SIGKDD Explorations 1 (2) (2000) 63–64.