MOVIH-IDS: A Mobile-Visualization Hybrid Intrusion Detection System

Álvaro Herrero[1], Emilio Corchado[1], María A. Pellicer[1], and Ajith Abraham[2]

[1] Department of Civil Engineering, University of Burgos

C/ Francisco de Vitoria s/n, 09006 Burgos, Spain

{ahcosio, escorchado}@ubu.es

[2] Centre for Quantifiable Quality of Service in Communication Systems

Norwegian University of Science and Technology

Trondheim, Norway

ajith.abraham@ieee.org

Abstract. A novel hybrid artificial intelligent system for Intrusion Detection, called MOVIH-IDS, is presented in this study. A hybrid model built by means of a multiagent system that incorporates an unsupervised connectionist Intrusion Detection System (IDS) has been defined to guaranty an efficient computer network security architecture. This hybrid IDS facilitates the intrusion detection in dynamic networks, in a more flexible and adaptable manner. The proposed improvement of the system in this paper includes deliberative agents characterized by the use of an unsupervised connectionist model to identify intrusions in computer networks. This hybrid IDS has been probed through several real anomalous situations related to the Simple Network Management Protocol as it is potentially dangerous. Experimental results probed the successful detection of such attacks through MOVIH-IDS.

Keywords: Hybrid Artificial Intelligent Systems, Multiagent Systems, Artificial Neural Networks, Unsupervised Projection Methods, Intrusion Detection Systems.

1 Introduction

As it is known, the rapid growing of computer networks and the interconnection among them has entailed some security problems. New security failures are discovered everyday and there are a growing number of hackers trying to take advantage of such failures. Intrusions are produced by attackers accessing to the system from external networks such as the Internet, for authorized users who attempt to obtain more privileges for which they are not authorized or authorized users who misuse the privileges given to them. The complexity increases in the case of distributed network-based systems and insecure networks. It is unquestionable that organizations need to protect their systems from these intruders and consequently, new network security tools are being developed. Firewalls are the most widely used tool of this kind but Intrusion Detection Systems (IDSs) are becoming more and more popular nevertheless [54]. IDSs are tools designed to monitor and analyse computer or network events in order to detect suspect patterns that may relate to a network or system attack. They have become a common complementary tool in the computer security infrastructure of most organizations.

Hybrid Artificial Intelligence Systems (HAISs) [44], [49], [59], [70] combine both symbolic and subsymbolic paradigms in order to build more robust and trustworthy problem-solving models. These systems are becoming more popular due to their ability to solve a wide range of complex real-world problems related to such aspects as imprecision, uncertainty or high dimensionality among others. These systems allow applying both the knowledge and the data to solve problems in a more interesting way. Due to all these reasons we consider the HAISs approach as an interesting frame for the design of IDS.

MOVIH-IDS, the proposed IDS, can be defined as a HAIS based on a dynamic multiagent architecture employing deliberative agents capable of learning and evolving with the environment. Some of the agents contained in this architecture are known as CBR-BDI agents [12] because they integrate the Case-Based Reasoning (CBR) paradigm and the BDI (Believes, Desires and Intentions) model. In the case of MOVIH-IDS, an unsupervised neural model is embedded in such agents (see Section 5.3 for further details) to perform network Intrusion Detection (ID). CBR-BDI agents [19] use CBR as a reasoning mechanism, which allows them to learn from initial knowledge, to interact autonomously with the environment as well as with users and other agents within the system, and to have a large capacity for adaptation to the needs of their surroundings.

Embedding Artificial Neural Networks (ANNs) in the deliberative agents of a dynamic Multiagent System (MASs) let us take advantage of some of the properties of ANNs (generalization and pattern recognition) and agents (reactivity, proactivity and sociability) at the same time, making the ID task possible. Additionally, MOVIH-IDS provides the network administrator with a mobile visualization based on unsupervised learning. Hence, this IDS is suitable to detect 0-day (previously unseen) attacks. That is, it is able to identify attacks that were not contained in the data used for training the neural model.

The overall architecture of the system as well as its different components are described as follows: Section 2 summarizes previous work on the ID field, Section 3 outlines the concept of CBR-BDI agent, Section 4 introduces the applied neural model, Section 5 describes the overall architecture of MOVIH-IDS, Section 6 describes the main achievements of the proposed IDS, and finally, Section 7 presents some conclusions and future work.

2 Previous Work on Intrusion Detection

A broad number of techniques have been used to build IDSs since the initial works on this field [6], [24]. Artificial Intelligence (AI) provides interesting solutions to this kind of problems; there have been some previous attempts to take advantage of AI techniques in the field of ID, as for example MAS [21], [34], [68]. CIDS (Cougaar-based IDS) [21] provides a hierarchical security agent framework, where a security node consists of four different agents (manager agent, monitor agent, decision agent and action agent) developed over the Cougaar framework [1]. It uses intelligent decision support modules to detect some anomalies and intrusions from user to packet level. The output of CIDS (generated by the Action Agent) consists on the environment status report (IDMEF format [22]) as well as recommendations of actions to be taken against the ongoing intrusive activities.

In [34], a general MAS framework for ID is proposed. Authors suggest the development of four main modules, namely the sniffing module (to be implemented as a simple reflex agent), the analysis module (to be implemented as an agent that keeps track of the environment to look at past packets), the decision module (to be implemented as a goal based agent to make the appropriate decisions) and the reporting module (to be implemented as a simple reflex agent to give out reports or logs).

An extension of the AAFID (Autonomous Agents For Intrusion Detection) architecture is proposed in [68] for building IDSs mainly composed of agents, filters, transceivers and monitors. This architecture does not detail the inner structure or mechanisms of the proposed agents.

From the mobile-agent approach, some works [23], [41] have been carried out. APHIDS [23] implements the distributed search and analysis tasks with mobile agents equipped with scripting capability to automate evidence gathering. Two different agent classes are proposed in [41]: monitoring agents (AM) and managing agents (AZ). AM observe the nodes, process captured information and draw conclusions for the evaluation of the current state of system security. AM agents can travel along the network to monitor different areas that may be at risk of attacks. On the other hand, AZ agents are responsible for creating profiles of attacks, managing AM agents and updating its database and ontology.

Considering all this previous work on agent-based ID, the main novelty of MOVIH-IDS is the inclusion of deliberative (CBR-BDI) agents in a specific IDS for packet ID through visualization based on neural models.

Still under the AI-based ID approach, some different machine learning models have been successfully applied for ID, as for example [17], [20], [43], [45], [60], [73]. Some other techniques and paradigms have been also applied to ID, such as Expert Systems [39], [46], [47], [63], [71], Evolutionary Computation [4], [67], CBR [26], [50], [62] or Fuzzy Logic [5]. Most of these works face ID from a classification standpoint; that is, labelling traffic as belonging to normal or anomalous classes. On the contrary, the IDS proposed in this work depicts every single packet to provide the network administrator with an intuitive snapshot of the network traffic. Hence, it is the network administrator's responsibility to consider an anomalous situation as an attack.

Additionally, some approaches combine different AI techniques (such as genetic algorithms and fuzzy logic [65], genetic algorithms and K-Nearest Neighbor (K-NN) [51] or K-NN and ANNs [40] among others) in order to face ID from a hybrid point of view. Some of the ideas in these works (such as providing intelligence to agents, combining symbolic and subsimbolic paradigms and some others) have been picked up and incorporated in the MOVIH-IDS formulation.

Apart from the AI solutions, there have been some other approaches such as the application of statistical [25], [48] and signature verification [58], [66] techniques. One of the main problems of these techniques is the absence of mechanisms to automatically "learn" patterns of 0-day attacks. Additionally, there are several IDSs that can generate different alarms when an anomalous situation occurs, however, they can not provide a general overview of what is happening in the network.

Various visualization techniques have been also applied in the field of IDSs [2], [31], [42] , [52], [53], [56] to tackle this issue by providing a visual depiction of the network topology, ID logs, network statistics or some other info. The proposed model goes further and offers a complete and more intuitive visualization of network traffic by depicting each simple packet and providing the network administrator with a snapshot of network traffic and protocol interactions to identify anomalous situations. The use of mobile devices in this proposal helps the administrator by providing more freedom and information at any time.

In comparison with previous work, the originality of MOVIH-IDS lies on the combination of different AI paradigms (MAS, ANNs and CBR embedded in agents) to visualize network traffic for ID at packet level.

3 CBR-BDI Architecture

The MObile-VIsualization Hybrid IDS (MOVIH-IDS) [16], [17], [36] has been designed to detect anomalous situations taking place in a computer network. CBR-BDI agents [19], [57] use CBR systems [3] as a reasoning mechanism, which allows them to learn from initial knowledge, to interact autonomously with the environment, users and other agents within the system, and to have a large capacity for adaptation to the needs of their surroundings.

CBR-BDI agents [12], [19], [57] provide reasoning based on previous experiences as CBR systems use memory to solve new problems. The main idea when working with CBR systems is the concept of case, that can be seen as a past experience described by the 3-tuple <Problem, Solution, Results>. A case is composed of a problem description (initial state), the solution applied to solve the problem (the actions executed in order to achieve the objectives) and the result obtained after applying the solution (the final state and the evaluation of the actions executed). As a consequence, a case base (memory) must be maintained to solve new problems. When a new problem is presented, the system executes a CBR cycle, composed of four sequential stages: Retrieve (those cases with the most similar problem description to the current problem are recovered from the cases memory), Reuse (the solutions corresponding to the similar cases retrieved in the previous stage are reused to construct a new solution), Revise (the application of the proposed solution is evaluated), and Retain (the agent saves the new experience for future reuse). MOVIH-IDS includes deliberative agents using a CBR architecture allowing them to respond to events, to take the initiative according to their goals, to communicate with other agents, to interact with users, and to make use of past experiences to find the best information to achieve goals, to forget obsolete cases and to incorporate new ones. The proposed CBR-BDI agents work at a high level with the concepts of Believes, Desires and Intentions (BDI) [11]. CBR-BDI agents have learning and adaptation capabilities, what facilitates their work in dynamic environments. Different models can be embedded in the four steps of the CBR reasoning process. In this work, these agents use a neural unsupervised model (see section 4) to identify intrusions in computer networks.

As an evolution of CBR-BDI, Case-Based Planning - BDI (CBP-BDI) agents [7] have already been used in this research. In keeping with the main idea behind CBR, case-based planning [33] is based on solving new planning problems by reusing past successful plans [69]. One of the key points in the CBP-BDI based planning is the notation used to represent the solution (the plans). A solution can be seen as a sequence of intermediate states transited to go from an initial state to the final state. States are usually represented as propositional logic sets. The set of actions can be represented as a set of operators together with an order relationship.

4 Unsupervised Learning

A connectionist approach appears consistent with the ID setting mainly because it allows a system to empirically learn the input-output relationship between raw traffic and its subsequent interpretation. This section presents the unsupervised learning model used within some of the MOVIH-IDS deliberative agents. Unsupervised learning was chosen in this study because in a real-life situation there is no target reference with which to compare the output of the neural network. The use of this kind of learning is very appropriate, for instance in the case of identifying previously unseen attacks (0-day attacks) [43] due to the generalization capabilities of ANNs.

As it is mentioned above, several attempts have been made to apply ANNs to the field of ID. Most of them are based on a classificatory standpoint. A different approach is followed in this research, in which the main goal is to provide the network administrator with a snapshot of the network traffic, not only to detect anomalous situations but also to visualize protocol interactions and traffic volume. A projection method employing unsupervised learning has been included in the deliberative CBR-BDI agents. This model and its basis are described in this section.

4.1 Exploratory Projection Pursuit

Exploratory Projection Pursuit (EPP) [27] is a statistical technique for solving the complex problem of identifying structure in high-dimensional data. It identifies low-dimensional data projections in which structure is identified by eye and requires an index of “interestingness” that measures the degree of interest exhibited by each projection. Subsequently, the data is transformed by optimizing this index in order to examine in greater detail the projections of highest interest. From a statistical point of view the most interesting directions are those which are as non-Gaussian as possible. This statistical technique may be implemented using connectionist models [14], [18], [29], [30], [37], [38].

4.2 Maximum Likelihood Hebbian Learning

Maximum Likelihood Hebbian Learning (MLHL) [18], [30], as a connectionist version of EPP, identifies interestingness by maximising the probability of the residuals under specific Probability Density Functions (PDFs) which are non-Gaussian. The classical statistical method of EPP [27] provides a linear projection of a data set onto a set of basis vectors which best reveal the interesting structure in data. MLHL identifies interestingness by maximising the probability of the residuals under specific PDFs which are non-Gaussian.

MLHL, based on the Negative Feedback Network (NFN) [28], associates an input vector, $\boldsymbol{x}\in\Re^N$, with an output vector, $\boldsymbol{y}\in\Re^M$, computed as:

$$y_i = \sum_{j=1}^{\mathbf{N}} W_{ij} x_j \text{ for each } i, i = 1 \ldots M \text{ .} \quad \textbf{(1)}$$

Where M is the dimensionality of y .

The activation (e_j) is then fed back through the same weights and subtracted from the input:

$$e_j = x_j - \sum_{i=1}^{M} W_{ij} y_i \text{ for each } j, j = 1 \ldots N \text{ .} \quad \textbf{(2)}$$

Where N is the dimensionality of x .

Finally, the weights are updated according to the learning rule:

$$\Delta W_{ij} = \eta . y_i . sign(e_j) | e_j |^p \text{ .} \quad \textbf{(3)}$$

Where p is the parameter related to the MLHL energy function that is used to match the PDF and the learning rule.

MLHL is a family of learning rules based on maximizing the likelihood of the residual from a NFN whenever such residuals are deemed to come from a distribution in the exponential family. The main advantage of this model is that by maximizing the likelihood of the residual with respect to the actual distribution, we are matching the learning rule to the PDF of the residual by applying different values of the p parameter specified in the learning rule. Furthermore, the nature of quantification of the interestingness is in terms of how likely the residuals are under a particular model of the PDF of the residuals. As with standard EPP, the data is sphered before applying this learning method.

4.3 A Cooperative Version of MLHL

Cooperative Maximum Likelihood Hebbian Learning (CMLHL) [13] extends the MLHL model by adding lateral connections, which have been derived from the Rectified Gaussian Distribution [64]. The resultant net can find the independent factors of a data set but does so in a way that captures some type of global ordering in the data set. So, the final CMLHL model can be described as follows:

Feed forward step: Equation (1).

Lateral activation passing: $y_i(t+1) = [y_i(t) + \tau(b - Ay)]^+$. **(4)**

Feed back step: Equation (2).

Weight change: Equation (3).

CMLHL was initially applied to the field of AI [13], [15] to identify local filters in space and time. Lateral connections were derived from the Rectified Gaussian distribution [64] which is a modification of the standard Gaussian distribution in which the variables are constrained to be non-negative, enabling the use of non-convex energy functions. The standard Gaussian distribution may be defined by:

$$p(y) = Z^{-1} \cdot e^{-\beta \cdot E(y)}. \quad \textbf{(5)}$$

$$E(y) = \frac{1}{2} \cdot y^T \cdot Ay - b^T \cdot y. \quad \textbf{(6)}$$

in which, the quadratic energy function $E(y)$ is defined by the vector b and the symmetric matrix A. The parameter $\beta = 1/T$ is an inverse temperature. Lowering the temperature concentrates the distribution at the minimum of the energy function. The Z factor normalizes the integral of $p(y)$ to unity.

The cooperative distribution is chosen as its modes are closely spaced along a non-linear continuous manifold. The energy functions that can be used are those that block the directions in which the energy diverges towards negative infinity. Thus, the matrix has to fit the following property:

$$y^T \cdot Ay > 0, \forall y : y_i > 0, i = 1 \ldots M. \quad \textbf{(7)}$$

Where M is the dimensionality of y.

The cooperative distribution in the case of N variables is defined by:

$$A_{ij} = \delta_{ij} + \frac{1}{N} - \frac{4}{N} \cdot \cos\left(\frac{2\pi}{N}(i-j)\right). \qquad \textbf{(8)}$$

$$b_i = 1. \qquad \textbf{(9)}$$

in which i and j are the output neuron identifiers and δ_{ij} is the Kronecker delta.

Matrix A is used to modify the response to the data based on the relation of the distances between the outputs. The projected gradient method is used [15], consisting of a gradient step followed by a rectification:

$$y_i(t+1) = \left[y_i(t) + \tau(b - Ay)\right]^+. \qquad \textbf{(10)}$$

in which the rectification $[\ ]^+$ is necessary to ensure that the y-values remain within the positive quadrant. If the step size (τ) is correctly chosen, this algorithm will probably be shown to converge to a stationary point of the energy function [9]. In practice, this stationary point is generally a local minimum.

The distribution mode can be approached by gradient descent on the derivative of the energy function with respect to y:

$$\Delta y \propto -\frac{\partial E}{\partial y} = -(Ay - b) = b - Ay. \qquad \textbf{(11)}$$

as used in (10).

The resulting model (CMLHL) can expose the independent factors of a data set in a way that captures some type of global ordering in the data set and displays it with greater sparsity than other models.

5 MOVIH-IDS

MOVIH-IDS is an IDS for distributed computer networks, incorporating different types of agents. Some of them are reactive agents while others are deliberative (CBR-BDI) agents.

An extended version of the Gaia methodology [72] has been applied to design this MAS. The following roles where identified after the Architectural Design Stage:

- SNIFFER: this role involves continuously capturing the traffic data flowing across a network segment. At the same time, when there is enough captured data, this data is split and its readiness is communicated to other roles.
- PREPROCESSOR: this role preprocesses the captured data. After that, an analysis for this new piece of data is requested.
- ANALYZER: this role negotiates for data analysis. Once an analysis is assigned to this role, it analyzes the new preprocessed data.
- CONFIGURATIONMANAGER: this role involves managing the configuration of several parameters (related to the splitting, pre-processing and the analysis of traffic data) and making such information available to some other roles.
- COORDINATOR: this organizational role involves coordinating some of the other roles and balancing the workload among them.
- VISUALIZER: this role is responsible for updating the visualization when new information (analyzed data or system information) is generated.

The following protocols have been also defined after this stage: AnalysisAborted, AnalysisCompleted, ChangeSplitConfig, ChangePreprocessConfig, ChangeAnalysisConfig, ManageSplitError, NegotiateAnalysis, PreprocessAborted, PreprocessedDataReady, RequestAnalysisConfig, RequestAnalyzedData, RequestPreprocessConfig, RequestPreprocessedData, RequestSplitConfig, RequestSplitData, RequestVisualization, SplitAborted, SplitDataReady, UpdateAnalysisConfig, UpdatePreprocessConfig, UpdateSplitConfig, UpdateSystemInfo.

The Detailed Design Stage concluded that there is a one-to-one correspondence between roles and agent classes in the system, as can be seen in the Agent Model (Fig. 1). The outcomes of Gaia methodology were modelled by Agent-UML (AUML) [8], that is a set of UML idioms and extensions for modelling agents.

<Figure 1 goes here>

Six agents (SNIFFER, PREPROCESSOR, ANALYZER, CONFIGURATIONMANAGER, COORDINATOR and VISUALIZER) have been developed in this study, as can be seen in Fig.2. They all are described placing special attention to the ANALYZER hybrid CBR-BDI agent, as it is the most complex one.

<Figure 2 goes here>

5.1 SNIFFER

This reactive agent is in charge of capturing traffic data. The continuous traffic flow is captured and split into segments (of preconfigured size) in order to send it through the network for further process. Finally, the readiness of the segmented data is communicated. One agent of this class is located in each of the network segments that the IDS has to cover (from 1 to *n*). Additionally, there are cloned agents (one per network segment) ready to substitute the active ones if they fail. These are the only backup instances in the whole system because these agents are the most critical ones. ID can not be performed if traffic data is not captured.

5.2 PREPROCESSOR

After splitting traffic data, the generated segments must be preprocessed to apply subsequent analysis. For the sake of network traffic, it would be advisable to locate one of these reactive agents in each host where SNIFFER agents are located. By doing so, just lightweight preprocessed data will travel along the network instead of the high-volume raw data. The sending of this data will not overload the network as its volume is much smaller than the one of split data. Once the data has been preprocessed, an analysis for this new piece of data is requested.

5.3 ANALYZER

As previously said, this is a hybrid CBR-BDI agent. It has got embedded the CMLHL (See Section 4.3) model within the adaptation stage of its CBR system that helps to analyze pre-processed traffic data. This agent generates a solution (or achieve its goals) by retrieving a previously analyzed case and analyzing the new one through the CMLHL architecture. Each case incorporates several features, as can be seen in Table 1.

<Table 1 goes here>

As it is known, the CBR life cycle consists of four steps: retrieval, reuse, revision and retention [3]. The techniques and tools used by the ANALYZER agent to implement these steps are described in the following paragraphs and depicted in Fig. 3.

<Figure 3 goes here>

Retrieval Stage: when a new analysis is requested, the ANALYZER agent tries to find the most similar case to the new one in the database. Associative Retrieval [69] based on Euclidean distance is used to find the most similar case in the multidimensional space defined by the main features characterizing each case (see problem description features in Table 1).

Reuse Stage: once the most similar case has been selected, its solution is reused. This solution consists of the values of the parameters used to train the CMLHL model (see solution description features in Table 1).

A set of trainings (for the CMLHL model with a combination of different parameter values varying in a specified range) is proposed by tacking into account the distance between the new case and the most similar one. That is, if they are very similar, a reduced set of trainings are going to be performed. On the contrary, if the most similar case is far away from the new one, a great number of trainings with very different parameter values are going to be generated.

Revision Stage: the CMLHL model is trained with the new dataset and the combination of parameters values generated in the reuse stage. When the new projections (the outputs of the CMLHL model for each combination) of the dataset are ready, they are shown to the human user (the network administrator typically) through the VISUALIZER agent. The user then chooses one of these projections as the best one; the one that provides the clearest snapshot of the traffic evolution.

Retention Stage: once the best projection is selected by the user, the ANALYZER agent stores the new case containing the problem-descriptor and the solution (parameter values used to generate the projection) in the case base for future reuse (See Table 1).

The ANALYZER is clearly the most resource-consuming class of agents. The amount of computational resources needed to analyze the data coming from different network segments is extremely high. To overcome this demand, ANALYZER agents can be located in high-performance computing clusters or in less powerful machines (as can be seen in Fig. 2). MOVIH-IDS can be adapted in this way to the resources available for ID.

The ANALYZER agent incorporates two different behaviours, namely “learning” and “exploitation”. Initially, during the set-up stage, this agent incorporates new knowledge (modelled as sets of problem/solution) into the case base as previously described. This learning behaviour is characterized by the 4 above mentioned CBR steps (retrieval, reuse, revision and retention). Once the case base is wide enough, the exploitation behaviour is started. From then on, the revision and retention steps of the CBR are not performed. When a new analysis request (problem) arrives, the ANALYZER agent retrieves the most similar case previously stored in the case base. Then, the weights contained in the solution are reused to project the new data. The other parameters of the neural model are not reused, as the neural network is not trained again.

5.4 ConfigurationManager

It is worth mentioning the importance of the configuration information. The processes of data capture, split, preprocess and analysis depends on the values of several parameters, as for example: packets to capture, segment length, features to extract... All this information is managed by the ConfigurationManager reactive agent, which is in charge of providing this information to the Sniffer, Preprocessor and Analyzer agents.

5.5 Coordinator

There is only one instance of the Coordinator agent, but there can be several Analyzer agents (from 1 to *m*) located in heterogeneous computers, following a cluster-like distribution. In order to improve the system efficiency and perform an almost real-time processing, the preprocessed data must be dynamically and optimally assigned. The Coordinator agent is in charge of allocating the pending analyses to the available Analyzer agents and is defined as a CBP-BDI agent [7]. Case-based planning [33] is based on solving new planning problems by reusing past successful plans [69]. Load balancing and scheduling in a cluster built of heterogeneous computers (as the one supporting MOVIH-IDS) is an interesting matter [10]. Intelligent agents have been previously applied to load balancing; in [61] a multiagent system is in charge of optimally load-balancing work among different agents based on purely local information. In the present study, general information about all the Analyzer agents is available for the Coordinator agent. In [55] a negotiation strategy was chosen to allocate jobs to different resources by means of a multiagent system. Aspects such as price, performance and quality of service are considered. Additionally, a CBR is used to predict how long a resource will take to execute a job. This can be guessed by taking into account the machine's performance in past cases of running such job at the considered resource.

The Coordinator agent plans to allocate an analysis to one of the Analyzer agents based on the following criteria:

- Location: Analyzer agents located in the network segment where the Visualizer or Preprocessor agents (if exist) are placed would be chosen.
- Available resources of the computer where each Analyzer agent is running. There must be taken into account not only the resources of the computer but also the use rate of them. Thus, the work load of computers must be measured.
- Analysis demands: amount and volume of data to be analysed.

- ANALYZER agents behaviour: as it is previously stated, these agents behave in a “learning” or “exploitation” way. The learning behaviour causes an ANALYZER agent to spend more time than the exploitation one.

As a computer network is an unstable environment, the availability of ANALYZER agents changes dynamically. As network links can stop working from time to time, the COORDINATOR agent should be able to re-allocate the analyses previously assigned to the ANALYZER agents located in the network segment that is down at the present time. An adaptation algorithm was designed to allow dynamic replanning in execution time.

The four well-known steps defining the CBP life cycle of this agent are defined as follows:

(Plan) Retrieval Stage: when a new pre-processed dataset is ready, an analysis is requested to the COORDINATOR agent. Associative retrieval is followed by taking into account the case/plan description shown in table 2.

<Table 2 goes here>

Reuse Stage: the retrieved plan is adapted to the new planning problem. The only restriction is that the analyses that are being performed (whose results have not been reported yet) can not be reassigned. The other (pending) ones can be reassigned in order to optimize the overall performance.

Revision Stage: the plan revision is performed in two stages: the planning failures are roughly identified by unexploited resources. That is, a planning failure is identified by the following situation: one of the ANALYZER agents is not performing any analysis and the other ones have a list of pending analysis. Additionally, execution failures are detected when communication with ANALYZER agents is interrupted. Information about these failures is stored in the case base (as shown in table 2) for future consideration. When an execution failure is detected, the CBR life cycle is run from the beginning. That is, the analysis request is considered new.

Retention Stage: when a plan is adopted, the COORDINATOR agent stores a new case containing the dataset-descriptor and the solution (See Table 2).

5.6 VISUALIZER

At the very end of the ID process, the analyzed data is presented to the network administrator (or the person in charge of the network) by means of a functional and mobile visualization through this interface agent. To improve the accessibility of the system, the administrator may visualize the results on a mobile device (as can be seen in Fig. 2), enabling informed decisions to be taken anywhere and at any time. Depending on the platform where the information will be shown, the offered visualization facilities will be different.

Instances of the VISUALIZER agent can be run in both computers and mobile devices (such as phones, PDAs, etc.). There are some restrictions derived from the use of mobile platforms as these platforms do not have as much resources as others have. That is the case of the Java 2 Platform - Micro Edition (J2ME), on which VISUALIZER agents are run over mobile devices. These agents can only provide a reduced set of interface features. Thus, two kinds of VISUALIZER agents are implemented:

– Mobile VISUALIZER agents: run on mobile platforms providing reduced interface features.

– Advanced VISUALIZER agents: run on platforms without interface limitations.

6 Outcome of MOVIH-IDS

Once traffic data is projected by MOVIH-IDS, “normal” traffic is depicted as parallel straight lines, as it is shown in Fig. 4. Hence, MOVIH-IDS identifies anomalous situations due to the fact that these situations do not tend to resemble parallel and smooth directions (as normal situations do) or due to their high temporal concentration of packets.

This can be seen in Fig. 4, where some UDP traffic has been captured and visualized in a mobile platform. It is easy to notice some different directions (Groups A and B) to the normal data ones, which evolve in the same and parallel direction. Also, the density of the packets is higher for these anomalous groups. All the packets contained in these anomalous groups are related to a transfer of quite dangerous information: and MIB Information Transfer [16], [17].

<Figure 4 goes here>

Another anomalous situation (Group C) can be identified in Fig. 4 due to its evolution in a direction non-parallel to the normal one. This group of packets is related to a port/network scan [16], [17].

To probe the effectiveness of the chosen projection model (See Section 4.3), it has been compared with other projection methods for data visualization, such as Principal Component Analysis (PCA) [17] and MLHL [35] as detailed in Section 6.1.

By providing an intuitive way of visualizing traffic data, this projection-based approach allows the packet-level analysis keeping sight of the big picture of the whole network. Knowing the network is a crucial issue in

ID [32] and MOVIH-IDS can help inexperienced network administrators to distinguish between normal and anomalous traffic.

6.1 Comparison

For comparison purposes, the same traffic dataset is projected by PCA, MLHL and CMLHL in Fig. 5. This dataset contains examples of 2 anomalous situations described in previous work [16]: a transfer of some information stored in the SNMP Management Information Base (Groups A and B in Fig. 5) and several port sweeps (Group C in Fig. 5).

<Figure 5 goes here>

As can be seen in Fig. 5.a, CMLHL is able to identify both anomalous situations while PCA (Fig.5.c) is just identifying the port sweeps (Group C) by means of normal/abnormal directions. Fig.5.b shows how MLHL is capable of detecting the MIB transfer (Groups A and B) but the port sweep (Group C) is not detected as clearly as by using CMLHL. CMLHL highlights anomalous situations more clearly because the projections are more spread out, easing the identification of anomalous groups. The anomalous situations are detected due to the different traffic directions or the high temporal density. CMLHL shows both features better than the other methods. MOVIH-IDS has been tested in facing some anomalous situations, and the effectiveness of the underlying neural model has been checked in previous works [16], [17], [36].

7 Conclusions and Future Work

This research presents a novel hybrid IDS whose main novelties, considering previous work, are:

- Inclusion of deliberative (CBR-BDI) agents in the proposed MAS.
- Application of unsupervised neural models for traffic data projection.
- Continuous inspection of network traffic by visualizing individual packets and not summarized info.

The hybrid approach and the features above mentioned provide MOVIH-IDS with the following advantages:

- Scalability: new agents (both SNIFFER and ANALYZER) can be dynamically added at any time.
- Failure tolerance: backup instances of some agents can be ready to run as soon as the working instances fail, showing a proactive behaviour.

- Real-time processing: by splitting the data and allowing the system to analyse it in different processing units (agents located in different machines).
- Mobile visualization: the visualization task can be performed in a wide variety of devices (as it is shown in Fig. 4).

Future work will focus on improving the dynamical assignment of analysis in order to take advantage of the computational resources in a more efficient way and on studying different distributions, learning rules, and neural models for the data analysis. Additionally, the upgrade of the COORDINATOR agent to deal with a high performance computing cluster is proposed.

Acknowledgments. This research has been partially supported by the project BU006A08 of the JCyL.

References

1. Cougaar: Cognitive Agent Architecture (http://cougaar.org/).
2. MRTG: The Multi Router Traffic Grapher (http://www.mrtg.org).
3. Aamodt, A., Plaza, E. Case-Based Reasoning - Foundational Issues, Methodological Variations, and System Approaches. AI Communications, Vol. 7(1), pp. 39-59 (1994).
4. Abraham, A., Grosan, C., Martin-Vide, C. Evolutionary Design of Intrusion Detection Programs. Int. Journal of Network Security, Vol. 4(3), pp. 328-339 (2007).
5. Abraham, A., Jain, R., Thomas, J., Han, S. Y. D-SCIDS: Distributed Soft Computing Intrusion Detection System. Journal of Network and Computer Applications, Vol. 30(1), pp. 81-98 (2007).
6. Anderson, J. P. Computer Security Threat Monitoring and Surveillance. Technical Report, Washington, PA, James P. Anderson Co (1980).
7. Bajo, J., Corchado, J., Rodríguez, S. Intelligent Guidance and Suggestions Using Case-Based Planning. Case-Based Reasoning Research and Development. pp. 389-403. Springer, Heidelberg (2007).
8. Bauer, B., Müller, J. P., Odell, J. Agent UML: A Formalism for Specifying Multiagent Software Systems. International Journal of Software Engineering and Knowledge Engineering, Vol. 11(3), pp. 1-24 (2001).
9. Bertsekas, D. P. Nonlinear Programming. ISBN: Athena Scientific Belmont, Mass (1999).
10. Bevilacqua, A. A Dynamic Load Balancing Method On A Heterogeneous Cluster Of Workstations. Informatica, Vol. 23(1), pp. (1999).
11. Bratman, M. E. Intentions, Plans and Practical Reason. ISBN: Harvard University Press, Cambridge, M.A. (1987).
12. Carrascosa, C., Bajo, J., Julián, V., Corchado, J. M., Botti, V. Hybrid Multi-agent Architecture as a Real-Time Problem-Solving Model. Expert Systems with Applications: An International Journal, Vol. 34(1), pp. 2-17 (2008).
13. Corchado, E., Fyfe, C. Connectionist Techniques for the Identification and Suppression of Interfering Underlying Factors. Int. Journal of Pattern Recognition and Artificial Intelligence, Vol. 17(8), pp. 1447-1466 (2003).
14. Corchado, E., Fyfe, C., Saiz, L., Lara, A. Development of a global and integral model of business management using an unsupervised model. IDEAL 2004. LNCS, Vol. 3177, pp. 499-504. Springer, Heidelberg (2004).
15. Corchado, E., Han, Y., Fyfe, C. Structuring Global Responses of Local Filters Using Lateral Connections. Journal of Experimental & Theoretical Artificial Intelligence, Vol. 15(4), pp. 473-487 (2003).
16. Corchado, E., Herrero, A., Saiz, J. M. Testing CAB-IDS through Mutations: on the Identification of Network Scans. KES 2006. LNCS (LNAI), Vol. 4252, pp. 433-441. Springer, Heidelberg (2006).
17. Corchado, E., Herrero, A., Sáiz, J. M. Detecting Compounded Anomalous SNMP Situations Using Cooperative Unsupervised Pattern Recognition. ICANN 2005. LNCS, Vol. 3697, pp. 905-910. Springer, Heidelberg (2005).
18. Corchado, E., MacDonald, D., Fyfe, C. Maximum and Minimum Likelihood Hebbian Learning for Exploratory Projection Pursuit. Data Mining and Knowledge Discovery, Vol. 8(3), pp. 203-225 (2004).
19. Corchado, J. M., Laza, R. Constructing Deliberative Agents with Case-Based Reasoning Technology. International Journal of Intelligent Systems, Vol. 18(12), pp. 1227-1241 (2003).
20. Chebrolu, S., Abraham, A., Thomas, J. P. Feature Deduction and Ensemble Design of Intrusion Detection Systems. Computers & Security, Vol. 24(4), pp. 295-307 (2005).
21. Dasgupta, D., Gonzalez, F., Yallapu, K., Gomez, J., Yarramsettii, R. CIDS: An agent-based intrusion detection system. Computers & Security, Vol. 24(5), pp. 387-398 (2005).

22. Debar, H., Curry, D., Feinstein, B. The Intrusion Detection Message Exchange Format (IDMEF). IETF RFC 4765. (2007).
23. Deeter, K., Singh, K., Wilson, S., Filipozzi, L., Vuong, S. APHIDS: A Mobile Agent-Based Programmable Hybrid Intrusion Detection System. Mobility Aware Technologies and Applications. LNCS, Vol. 3284, pp. 244-253. Springer, Heidelberg (2004).
24. Denning, D. E. An Intrusion-Detection Model. IEEE Transactions on Software Engineering, Vol. 13(2), pp. 222-232 (1987).
25. DuMouchel, W., Schonlau, M. A comparison of test statistics for computer Intrusion detection based on principal components regression of transition probabilities. Proc. of the 30th Symposium on the Interface: Computing Science and Statistics. Proc. of the 30th Symposium on the Interface: Computing Science and Statistics, pp. 404-413 (1999).
26. Esmaili, M., Balachandran, B., Safavi-Naini, R., Pieprzyk, J. Case-Based Reasoning for Intrusion Detection. Proceedings of the 12th Annual Computer Security Applications Conference. IEEE Computer Society (1996).
27. Friedman, J. H., Tukey, J. W. A Projection Pursuit Algorithm for Exploratory Data-Analysis. IEEE Transactions on Computers, Vol. 23(9), pp. 881-890 (1974).
28. Fyfe, C. A Neural Network for PCA and Beyond. Neural Processing Letters, Vol. 6(1-2), pp. 33-41 (1997).
29. Fyfe, C., Baddeley, R., McGregor, D. R. Exploratory Projection Pursuit: an Artificial Neural Network Approach. Research Report/94/160, University of Strathclyde. (1994).
30. Fyfe, C., Corchado, E. Maximum Likelihood Hebbian Rules. Proc. of the 10th European Symposium on Artificial Neural Networks (ESANN 2002). pp. 143-148 (2002).
31. Goldring, T. Scatter (and Other) Plots for Visualizing User Profiling Data and Network Traffic. Proc. of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security. pp. 119-123. ACM Press (2004).
32. Goodall, J. R., Lutters, W. G., Komlodi, A. The Work of Intrusion Detection: Rethinking the Role of Security Analysts. Proceedings of the Americas Conference on Information Systems. pp. 1421–1427 (2004).
33. Hammond, K. J. Case-based Planning: Viewing Planning as a Memory Task. ISBN: 0-12-322060-2. Academic Press Professional, Inc. (1989).
34. Hegazy, I. M., Al-Arif, T., Fayed, Z. T., Faheem, H. M. A Multi-agent Based System for Intrusion Detection. IEEE Potentials, Vol. 22(4), pp. 28-31 (2003).
35. Herrero, A., Corchado, E., Gastaldo, P., Zunino, R. A Comparison of Neural Projection Techniques Applied to Intrusion Detection Systems. IWANN'2007. LNCS, Vol. 4507, pp. 1138-1146. Springer, Heidelberg (2007).
36. Herrero, A., Corchado, E., Sáiz, J. M. MOVICAB-IDS: Visual Analysis of Network Traffic Data Streams for Intrusion Detection. IDEAL 2006. LNCS, Vol. 4224, pp. 1424-1433. Springer, Heidelberg (2006).
37. Hyvarinen, A. Complexity pursuit: Separating interesting components from time series. Neural Computation, Vol. 13(4), pp. 883-898 (2001).
38. Hyvärinen, A. New Approximations of Differential Entropy for Independent Component Analysis and Projection Pursuit. Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems. pp. 273 - 279. MIT Press (1998).
39. Javitz, H. S., Valdes, A. The SRI IDES Statistical Anomaly Detector. IEEE Computer Society Symposium on Research in Security and Privacy, 1991. pp. 316-326 (1991).
40. Kholfi, S., Habib, M., Aljahdali, S. Best Hybrid Classifiers for Intrusion Detection. Journal of Computational Methods in Science and Engineering, Vol. 6(2), pp. 299 - 307 (2006).
41. Kolaczek, G., Pieczynska-Kuchtiak, A., Juszczyszyn, K., Grzech, A., Katarzyniak, R. P., Nguyen, N. T. A Mobile Agent Approach to Intrusion Detection in Network Systems. Knowledge-Based Intelligent Information and Engineering Systems. LNAI, Vol. 3682, pp. 514-519. Springer Heidelberg (2005).
42. Kulsoom, A., Lee, C., Conti, G., Copeland, J. A. Visualizing Network Data for Intrusion Detection. Proc. of the Sixth Annual IEEE Information Assurance Workshop - Systems, Man and Cybernetics (SMC) , 2005. pp. 100-108 (2005).
43. Laskov, P., Dussel, P., Schafer, C., Rieck, K. Learning Intrusion Detection: Supervised or Unsupervised? ICIAP 2005. LNCS, Vol. 3617, pp. 50-57. Springer, Heidelberg (2005).
44. Lees, B. Hybrid Case-Based Reasoning Systems. Proceedings of the ICCBR'99 Workshop. Vol. 99, (1999).
45. Liao, Y. H., Vemuri, V. R. Use of K-Nearest Neighbor Classifier for Intrusion Detection. Computers & Security, Vol. 21(5), pp. 439-448 (2002).
46. Lindqvist, U., Porras, P. A. Detecting Computer and Network Misuse through the Production-based Expert System Toolset (P-BEST). IEEE Symposium on Security and Privacy, 1999. pp. 146-161 (1999).
47. Lunt, T. F. IDES: An Intelligent System for Detecting Intruders. Proceedings of the Symposium: Computer Security, Threat and Countermeasures. (1990).
48. Marchette, D., J. Computer Intrusion Detection and Network Monitoring: A Statistical Viewpoint. Information Science and Statistics. ISBN: 0387952810. Springer-Verlag New York, Inc. (2001).
49. Medsker, L. R. Hybrid Intelligent Systems. ISBN: 0792395883. Kluwer Academic Publishers (1995).
50. Micarelli, A., Sansonetti, G. A Case-Based Approach to Anomaly Intrusion Detection. Machine Learning and Data Mining in Pattern Recognition. LNCS, Vol. 4571, pp. 434-448. Springer, Heidelberg (2007).
51. Middlemiss, M., Dick, G. Feature Selection of Intrusion Detection Data Using a Hybrid Genetic Algorithm/KNN Approach. Design and Application of Hybrid Intelligent Systems. IOS Press. Design and Application of Hybrid Intelligent Systems, pp. 519-527. IOS Press (2003).

52. Mizoguchi, F. Anomaly Detection using Visualization and Machine Learning. Proeedings of the IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2000. (WET ICE 2000). . pp. 165-170 (2000).
53. Muelder, C., Ma, K. L., Bartoletti, T. Interactive Visualization for Network and Port Scan Detection. RAID 2006. LNCS, Vol. 3858, pp. 265-283. Springer, Heidelberg (2006).
54. Mukherjee, B., Heberlein, L. T., Levitt, K. N. Network Intrusion Detection. Network, IEEE, Vol. 8(3), pp. 26-41 (1994).
55. Noronha Nassif, L., Marcos Nogueira, J., Vinicius de Andrade, F., Ahmed, M., Karmouch, A., Impey, R. Job Completion Prediction in Grid using Distributed Case-based Reasoning. Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise. pp. 249-254 (2005).
56. Nyarko, K., Capers, T., Scott, C., Ladeji-Osias, K. A. Network Intrusion Visualization with NIVA, an Intrusion Detection Visual ANALYZER with Haptic Integration. Proceedings of the 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2002. (HAPTICS 2002). pp. 277-284 (2002).
57. Pellicer, M. A., Corchado, J. M. Development of CBR-BDI Agents. International Journal of Computer Science and Applications, Vol. 2(1), pp. 25 - 32 (2005).
58. Roesch, M. Snort–Lightweight Intrusion Detection for Networks. Proc. of the 13th Systems Administration Conf. (LISA '99). Proc. of the 13th Systems Administration Conf. (LISA '99), pp. 229-238 (1999).
59. Ron, S., Frederic, A. Connectionist-Symbolic Integration: From Unified to Hybrid Approaches, Lawrence Erlbaum Associates, Inc.
60. Sarasamma, S. T., Zhu, Q. M. A., Huff, J. Hierarchical Kohonenen Net for Anomaly Detection in Network Security. IEEE Transactions on Systems Man and Cybernetics, Part B, Vol. 35(2), pp. 302-312 (2005).
61. Schaerf, A., Shoham, Y., Tennenholtz, M. Adaptive Load Balancing: A Study in Multi-Agent Learning. Journal of Artificial Intelligence Research, Vol. 2, pp. 475-500 (1995).
62. Schwartz, D. G., Stoecklin, S., Yilmaz, E. A Case-Based Approach to Network Intrusion Detection. Proceedings of the Fifth International Conference on Information Fusion, 2002. Vol. 2, pp. 1084-1089 (2002).
63. Sebring, M., Shellhouse, E., Hanna, M., Whitehurst, R. Expert Systems in Intrusion Detection: A Case Study. Proceedings of the 11th National Computer Security Conference. pp. 74-81 (1988).
64. Seung, H. S., Socci, N. D., Lee, D. The Rectified Gaussian Distribution. Advances in Neural Information Processing Systems, Vol. 10, pp. 350-356 (1998).
65. Sindhu, S. S. S., Ramasubramanian, P., Kannan, A. Intelligent Multi-agent Based Genetic Fuzzy Ensemble Network Intrusion Detection. Neural Information Processing. LNCS, pp. 983-988. Springer, Heidelberg (2004).
66. Sommer, R., Paxson, V. Enhancing Byte-Level Network Intrusion Detection Signatures with Context. Proc. of the 10th ACM Conf. on Computer and Communications Security. pp. 262-271. ACM Press (2003).
67. Song, D., Heywood, M. I., Zincir-Heywood, A. N. A Linear Genetic Programming Approach to Intrusion Detection. Proceedings of the Genetic and Evolutionary Computation - Gecco 2003. LNCS, Vol. 2724, pp. 2325-2336. Springer, Heidelberg (2003).
68. Spafford, E. H., Zamboni, D. Intrusion Detection Using Autonomous Agents. Computer Networks: The Int. Journal of Computer and Telecommunications Networking, Vol. 34(4), pp. 547-570 (2000).
69. Spalzzi, L. A Survey on Case-Based Planning. Artificial Intelligence Review, Vol. 16(1), pp. 3-36 (2001).
70. Tran, C., Abraham, A., Jain, L. Decision Support Systems using Hybrid Neurocomputing. Neurocomputing, Vol. 61, pp. 85-97 (2004).
71. Vaccaro, H. S., Liepins, G. E. Detection of Anomalous Computer Session Activity. Proceedings of the 1989 IEEE Symposium on Security and Privacy. pp. 280-289 (1989).
72. Zambonelli, F., Jennings, N. R., Wooldridge, M. Developing Multiagent Systems: the Gaia Methodology. ACM Transactions on Software Engineering and Methodology, Vol. 12(3), pp. 317-370 (2003).
73. Zanero, S., Savaresi, S. Unsupervised Learning Techniques for an Intrusion Detection System. Proc. of the ACM Symposium on Applied Computing. pp. 412-419 (2004).

$$\textsc{Sniffer}^{1\ldots N} \xrightarrow{\textit{plays}} \textsc{Sniffer}$$

$$\textsc{Preprocessor}^{+} \xrightarrow{\textit{plays}} \textsc{Preprocessor}$$

$$\textsc{Analyzer}^{+} \xrightarrow{\textit{plays}} \textsc{Analyzer}$$

$$\textsc{ConfigurationManager}^{1} \xrightarrow{\textit{plays}} \textsc{ConfigurationManager}$$

$$\textsc{Coordinator}^{1} \xrightarrow{\textit{plays}} \textsc{Coordinator}$$

$$\textsc{Visualizer}^{+} \xrightarrow{\textit{plays}} \textsc{Visualizer}$$

Fig. 1. Agent model.

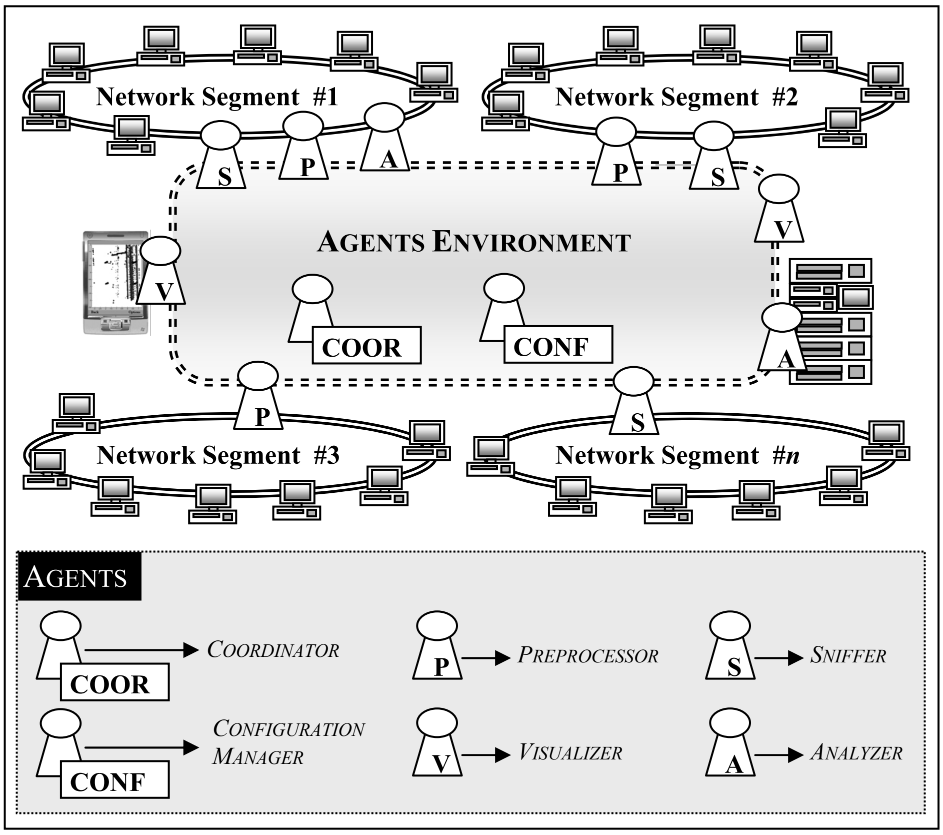


Fig. 2. MOVIH-IDS.

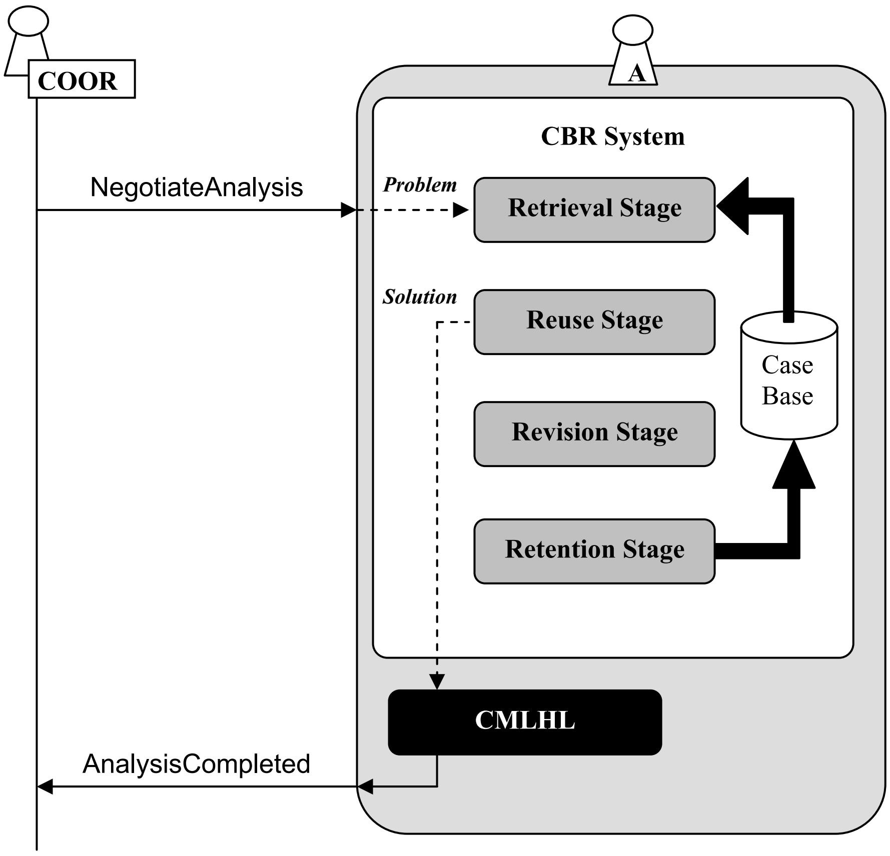


Fig. 3. ANALYZER agent architecture.

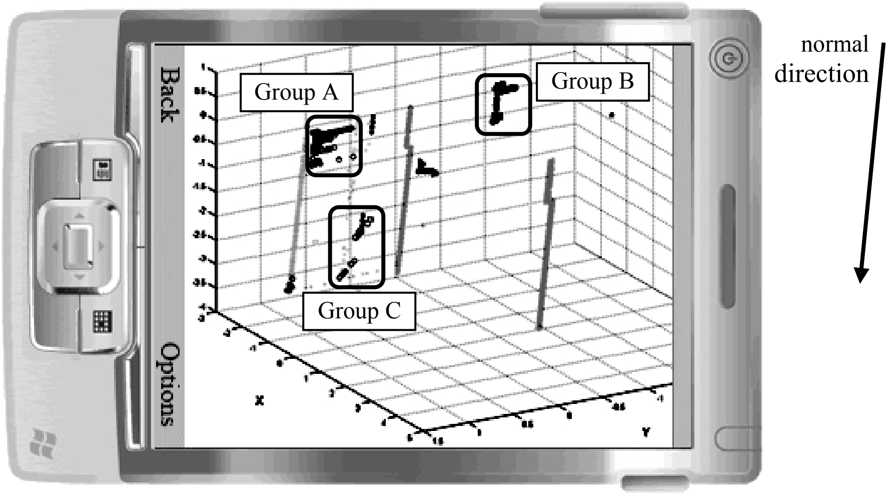
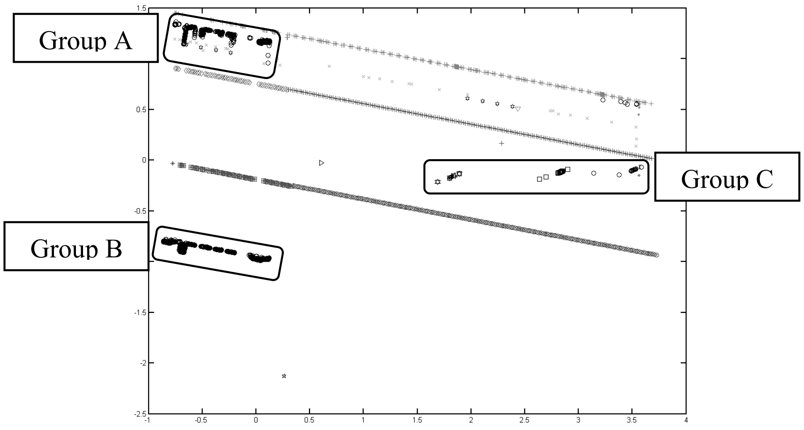
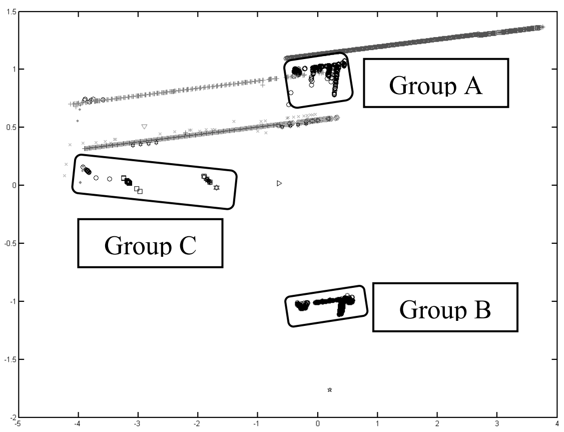
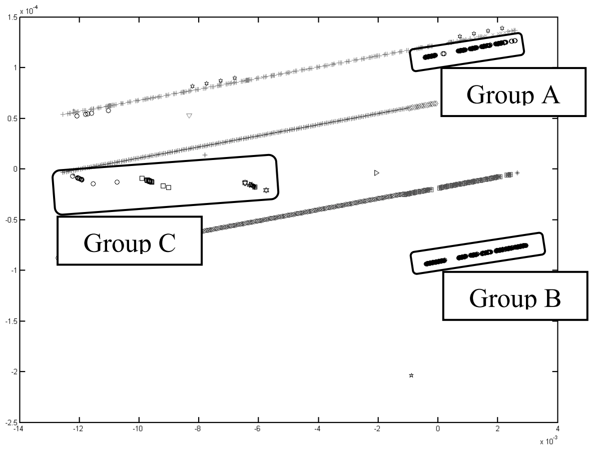


Fig. 4. Mobile visualization.



a) CMLHL





b) MLHL

c) PCA

Fig. 5. Comparative projections.

Table 1. ANALYZER Agent - Representation of case features. Classes: P (Problem description attribute) and S (Solution description attribute).

Class	**Feature**	**Type**	**Description**
P	Segment length	Integer	Total segment length (in ms).
P	Network segment	Integer	Network segment where the traffic comes from.
P	Date	Date	Date of capturing.
P	#source ports	Integer	Total number of source ports.
P	#destination ports	Integer	Total number of destination ports.
P	#protocols	Integer	Total number of protocols.
P	#packets	Integer	Total number of packets.
P	Protocol/packets	Array	An array (of variable length depending on each dataset) containing information about how many packets of each protocol there are in the dataset.
S	#Iterations	Integer	Number of iterations.
S	Learning rate	Float	Learning rate.
S	p	Float	CMLHL parameter.
S	Lateral strength	Float	CMLHL parameter.
S	Weights	Matrix	A matrix containing the synaptic weights calculated by the CMLHL model after training.

Table 2. COORDINATOR Agent - Representation of case features. Classes: P (Problem description attribute) and S (Solution description attribute).

Class	**Feature**	**Type**	**Description**
P	#packets	Integer	Total number of packets contained in the dataset to be analysed.
P	Analyzers / location	Array	An array (of variable length depending on the number of available ANALYZER agents) indicating the network segment where the ANALYZER agent is located.
P	Analyzers / features	Array	An array (of variable length depending on the number of available ANALYZER agents) containing information about the resources, their availability and pending tasks.
P	Analyzers / failures	Array	An array (of variable length depending on the number of available ANALYZER agents) containing information about the number of times each ANALYZER agent has stopped working in the recent past (execution failures).
S	Analyzers / plans	Array	An array (of variable length depending on the number of available ANALYZER agents) containing the analyses assigned to each ANALYZER agent.