Combining Techniques from Intelligent and Decision Support Systems: An Application in Network Security

Thomas Spyrou[1] & Rainer Telesko[2]

[1]University of the Aegean
Department of Mathematics
Research Laboratory of Samos
GR 83200 Karlovassi, Samos, Greece
e-mail: tsp@aegean.gr

[2]University of Vienna
Institute for Applied Computer Science
Department of Knowledge Engineering
AT 1210 Vienna, Austria
e-mail: telesko@dke.univie.ac.at

Abstract

This paper presents the design and development of a prototype of an expert system application for the detection of certain types of abnormal behaviour in open networks (and in particular in the management systems of such networks) and shows how it provides decision aid for the System Security Officer (SSO). The objective is to aid the SSO determine the seriousness of the possible attack and also to help him to find and apply the appropriate countermeasures. The case of unknown attacks is discussed here, where the SSO has to consider a number of constraints. Sets of actions acceptable by the system but which at a higher level may form a non acceptable task or tasks are assumed to be detected by the User Intention Identification (UII) module as potentially malicious intentions. The Decision Module (DM) is, responsible for aiding SSO with his decision regarding possible intrusion detected by the UII module. These two modules are presented and discussed and the way these two modules communicate is also introduced. This work is carried out under SECURENET, a project that aims at the protection of networks and in particular their management. The development of the current version of the modules was carried out in the frame-based expert system-shell CLIPS

Acknowledgement: The work reported on in this paper was funded by the RACE - SECURENET II (R2113) project.

1. Introduction

This paper presents the design and development of a prototype of an expert system application for the detection of certain types of abnormal behaviour in open networks (and in particular in the management systems of such networks) and shows how it provides decision aid for the System Security Officer (SSO). This work is being carried out under SECURENET [7,8], a project that aims at the protection of networks and in

particular their management. Such systems are open to various types of malicious attacks and intrusions which in most cases consist of a few or many illegal acts, and their detection usually triggers appropriate countermeasures. The present work is concerned with detection of that type of attacks/intrusions which do not usually consist of illegal actions, but of a set of actions acceptable by the system, but which at a higher level may form a non acceptable task or tasks. This form of intrusion is regarded as part of users' intentions about the use of the system, i.e. the tasks they intend to perform.

The objective is to aid the SSO determine the seriousness of the possible attack and also to help him to look for the appropriate countermeasures. This is fairly straightforward when a known attack is the issue but in the case of unknown attacks the SSO has to consider a number of constraints.

Two modules of the system are presented and discussed, the User Intention Identification (UII) module [9] and the Decision Module (DM) [3] and the way these two modules communicate is described.

The development of the current version of the modules was carried out in the frame-based expert system-shell CLIPS [2,4].

Section 2 gives an overview of the SECURENET system and its architecture. In section 3 and 4, surveys of the User Intention Identification module and the Decision module are given. Section 5 discusses the way these two modules are combined to produce a decision aid for the Security Officer of a network system and finally a section with conclusions and discussion follows.

2. An overview of SECURENET.

The SECURENET-project is developing a network-monitoring and analysis-system which aims at networks protection. It is composed of several modules with different responsibilities. According to the SECURENET II-architecture there are three necessary steps a system must perform in order to detect malicious attacks in a network: the first step where the network activity is monitored, the second step where this monitored activity is analysed according to a number of techniques, and the third stage where the results of this analysis are evaluated and the seriousness of the danger is estimated. If there is much evidence that there is an (ongoing) intrusion, appropriate countermeasures will be selected in the Countermeasure module - and when there is the 'O.K.' from the SSO - executed.

Figure 1 presents the SECURENET II architecture with the main modules and the respective interfaces.

The heart of the SECURENET-system according to this architecture is the analysis module which gathers information coming from the actual network in order to recognise or to infer malicious attacks to the network. Currently there are three analysis modules, the Neural Network module, the Expert System module and the User Intention Identification module. A Decision support module is also necessary in order to further elaborate the results of the analysis made by the analysis modules.

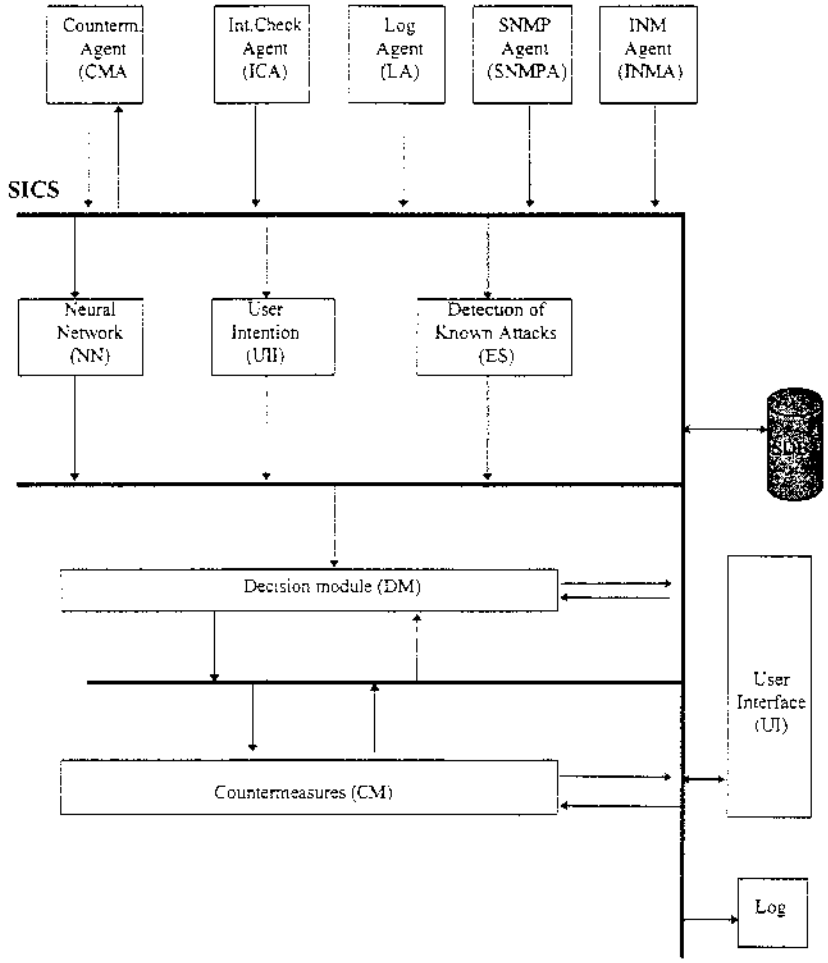

Figure 1. SECURENET II architecture

Finally there are several monitoring modules which observe the real network acquiring the information necessary for the analysis step. All these modules need to pass information each other in an efficient way. An intermodule communication system (SICS) [10,11,12] is used to play the role of communication manager and in parallel it has the responsibility to maintain a common communication dictionary.

3. Overview of the User Intention Identification Module (UII)

The User Intention Identification module is an autonomous module for the detection of anomalous behaviour by reasoning about the characterisation of the intentions of users. This module views the users of a system as using it in order to achieve certain goals by performing various tasks. This module plays a complementary role within the SECURENET-system, trying to detect a range of malicious attacks with special characteristics. The major characteristics of the malicious attacks that this module aims at, are those cases where malicious tasks are composed of legal events. Since the basic actions for these tasks are allowable they cannot be detected by simple matching mechanisms. The examination of the whole rationality behind the execution of these basic actions has to be considered in relation to the general goals these actions are trying to fulfil (when composed to form tasks). Reasoning about the deviations observed in the

execution of actions within a task in relation to the normal task execution (under the general goal-oriented constraints) offers an indication of the suspiciousness of the observed behaviour. This reasoning mechanism is based on a semantic matching of observed user activity with a representation of normal behaviour. Task related Knowledge structures (TKS) is utilised here for the representation of the normal behaviour of the users of a system when they perform a number of tasks.

In other words the UII system aims to detect a certain type of malicious attacks by characterising the normality of the behaviour of the users. Based on the knowledge representations described above the system represents the expected normal behaviour and compares it against the current, observed behaviour. Deviations correspond to abnormality and a certainty factor is calculated as an indication of the importance of this abnormality.

As a system, UII receives as input the action that the user is executing, the time of execution and the user identity and gives as output an indication of the suspiciousness of the observed behaviour and an evaluation of executed tasks.

Figure 2 gives an outline of the architecture of the User Intention Identification Module and its components which are discussed in the sequel.

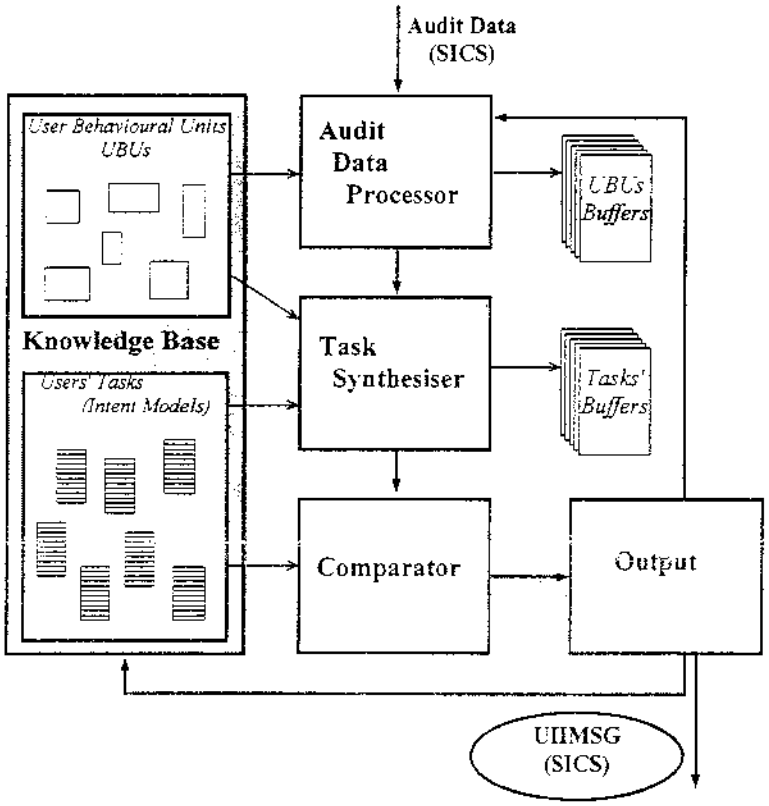

Figure 2. Architecture of the UII module

3.1 Task and Functionalities

The main objective of the User Intention Identification Module is the semantic

interpretation of the collected data that correspond to the network users' actions. According to this the Intention Identification module must be able to perform four major tasks.

- Pre-processing of the input data
- Task Synthesis.
- Behaviour Comparison.
- Output

There are four functions corresponding to these four tasks:

3.1.1 Audit Data Processor (ADP)

The input data for the UII is received/provided by the audit file. This audit file remains open during the process of identification of the tasks for execution (or for possible execution) by the UII. This file receives information about the user identity, the actions are executed in the net; this process is continuity. Every time that an identification has finished, a new audit is ready to be read by the UII. Sets of rules are triggered to determine the task or tasks to which this observed action can be attached.

3.1.2 Task Synthesiser

This function implements the first steps of the semantic interpretation of the behaviour of the observed network users. This interpretation is the basic function of the Intention Identification module. TKS entities identified by the ADP function are characterised as task components. Sets of rules are triggered to determine the task(s) to which these TKS entities can be attached. These rules represent the relations of the observed actions within and between the possible tasks executed by the observed entity. These relations are represented as pre-conditions or post-conditions, sequence relationships, association to wider tasks and goals etc.

The basic idea behind this function is that a first comparison has to be made in order to discard non valid hypotheses about the execution of tasks. This is necessary in order to let the comparator function work with valid tasks only. The examination of the TKS entities in this function is made primarily for the validation of an action as a part of a task.

3.1.3 Comparator

This is the main inference mechanism of the module. It makes all the semantic interpretation of the observed network user behaviour and produces the intrusion hypotheses. The basic idea behind this function is that by reasoning about deviations from the normal task execution and by reasoning about the similarities of the executed with allowable tasks, estimations regarding the suspiciousness of the performed activity could be made. These hypotheses are formed in the output function.

The reasoning mechanism of this function is realised with sets of rules that represent the relationships between the various parts of a TKS. The TKS knowledge structure is used as the knowledge base for that function and knowledge about task execution is represented with that knowledge structure. There are three major aims for that function:

- It tries to combine the relations between the TKS entities in order to decide about the

normality and validity in a Task execution

- It tries to combine inter-task relations and associations in relation to relevant goals in order to decide whether a combination of task execution is suspicious and finally
- It tries to combine the goal substructure of the various TKS structures of tasks executed with within and between role relations in the tasks execution. This is the most complex aim of the comparator function and for the purposes of the demonstrator the basic role relations will be examined.

The basic operation of this function is performed by a recursive process which elaborates the TKS structure representation in relation to the asserted facts that correspond to observed behaviour and come from the previous two functions.

The output of this function is a hypothesis that classifies a suspicious situation observed and provides the data for a representative description of the reasoning that produced that hypothesis.

3.1.4 Output

This function plays the role of the explanation generation part of an expert system. Every time that a hypothesis is generated by the comparator function this function selects the necessary information relevant to this suspected intrusion from the TKS based profile and offers it as a means of explanation. This information is passed to the Decision Module.

4. Overview of the Decision Module (DM)

The main task of the DM [3] is to produce a final decision about whether an attack has taken place or not. Figure 3 presents an outline of its internal architecture.

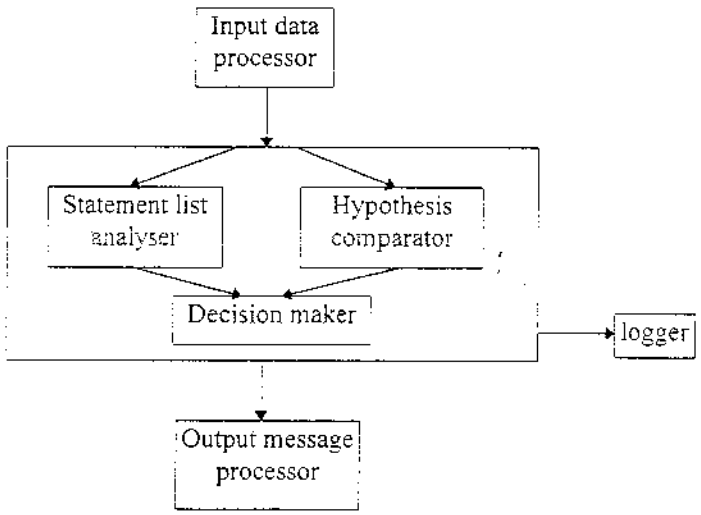

Figure 3. The architecture of Decision Module

The input to the DM are various hypotheses and confidence values delivered from the

- Expert System module (ES)
- Neural Net module (NN) and

- User Intention Identification module (UII).

4.1 Input Data Processor

The DM is triggered if at least one of the three detection modules (ES, NN or UII) sends a messages to the DM. The messages from the three detection modules are collected in a queue according to the FIFO-principle and have all the same general format (Table 1).

Field	Format	Description
system_id	string[8]	Operating system identifier
node_id	long	node identifier number
time	long	record producing time/date
record_id	long	serial number for record
user_id	long	unix audit user identifier
hypothesis_type	long	type of hypothesis
level_of_conf	long	level of confidence
param_number	long	number of additional parameters
params	char array	parameters

Table 1: General format of ES/NN/UII-messages

Most important for the DM are the fields *hypothesis_type*, which contains the generated hypothesis, *params*, which gives a list of the 'statements' that led to the alarm in the respective module and *level_of_conf*, which associates a confidence value between [0,...,9] for that hypothesis. '0' means no confidence, while a value of '9' shows high confidence in the generated hypothesis. The task of the DM is now to *confirm* or *reject* the previous generated hypothesis. In the first case the `level_of_conf` will be increased, in the latter case decreased. If there is no additional information available in the DM, the `level_of_conf` will remain unchanged. Two methods are available for performing this task:

4.2 Analysing the Statement list:

First of all one has to verify that the alarm raised by a module is not a 'false alarm'.

A one possible way to do this is simply analysing the statement-list (user commands). For example, if some critical statements (e.g. open file, write file, chmod, rm) or a critical 'combination' of statements is found, the `level_of_conf` will remain unchanged if not raised.

4.3 Hypothesis comparator:

If two or more alarms are raised in the same time period (this can be checked by 'temporal reasoning'), the highest `level_of_conf` will be delivered to the SSO and the Countermeasure module in order to make sure that the problem is not underestimated.

The field `attack classes` will be determined by matching the facts with previously defined rules. These rules are related to the targets of an intrusion and are domain-dependent. A small example is shown in section 5 ('reroute example').

4.4 Output message processor:

The output of the Decision Module is coded according to the DMMSG format (Table 2). This output is communicated with the Countermeasure Module and it is also passed to the System Security Officer.

Field	Format	Description
system_id	string[8]	Operating System-identification
node_id	long	node identifier number
time	long	record producing time/date
record_id	long	serial number for record
user_id	long	user identifier
level_of_conf	long	level of confidence
level_penet	int	level of penetration
attack_classes	long	attack classification
param_number	long	number of additional parameters
params	char array	parameters

Table 2: Format of DMMSG

The most important fields for the SSO here are the previous mentioned level_of_conf and the attack classes. The latter field (Table. 3) contains information (on a high level), about which sort of intrusion the SSO and the Countermeasure Module have to face with.

Attack class	**Symptoms**
No Attack	Used only by the DM for the report and log file.
Trojan Horse	Unexpected file operations, inappropriate source code, unexpected communications
Logic Bomb	Inappropriate source code, operations on sensitive resources
Insider attack	user operates outside user model thresholds, operations on sensitive resources, inappropriate creation and manipulation of data instances
Password Cracking	Repeated failed attempts to establish another identity. operations on passwd-file .
System programming attacks	attempt to exploit known weaknesses in a system routine
Outsider access violation	Repeated failed access to establish an identity; attempts to use default system passwords
Denial of service	Unexplained loss of contact with victim system, large amounts of meaningless traffic jamming available bandwidth
Trapdoor attack	inappropriate source code, operations on sensitive resources
Known attack	Use of known system bugs and -loopholes

Table 3: Attack classes in SECURENET

5. Producing useful aid.

This section describes the way the two modules UII and DM communicate information in order to produce a useful decision for the security officer. In other words the collaboration of a module that analyses observed behaviour and a module that utilises this information for use in the real world.

The whole functionality of the communication will be shown through an example where a produced hypothesis by the UII module triggers the DM.

5.1 Checking the UII-status:

First of all, the current state of the UII module has to be examined in order to estimate the weight of the results of the analysis.

A fuzzy variable called `uii_state` to hold the current status of UII is introduced:

```
uii_state = {UNSTABLE, WEAK, NORMAL}
```

`uii_state` has to be set to UNSTABLE if UII is out of operation (e.g. the knowledge base is updated with new tasks). `uii_state` will be WEAK if UII has produced an unusual number of false alarms in the past. If UII works quite normally, then `uii_state` gets a NORMAL.

The value of `uii_state` is stored in the DM configuration file, which will be read anytime the DM is started and updated and anytime the DM terminates. Furthermore the SSO will be informed of the actual entry of `uii_state`. For example if there is a WEAK and there comes an alarm from the other analysis modules related to the same time period, this has the practical consequence that UII has not the same 'weight' in the subsequent decision process. The more false alarms are produced the less will be the weight of this module.

5.2 Analysing UII-hypothesis:

Let us assume that `uii_state` has a NORMAL and the following example-hypothesis is delivered within an UII-message to DM:

Hypothesis	Description of Hypothesis	Parameters
600400	The hypothesis is produced when there is strong evidence from the history of the user that during the execution of a number of tasks there exist crucial strategy or goal conflicts. These conflicts are identified after the comparison of the current history of the user with the knowledge in the knowledge base about the tasks/goals relations	Possible executed tasks, conflicted strategies/goals

Example hypothesis:

(Only relevant information is given below)

```
Hypothesis-number (rule set) : 600400
level_of_conf: 5
task: remove a routing node for service
strategy: 1) check system state
          2) reroute traffic
          3) remove routing node
          4) check system state
params:   reroute traffic
# params: 1
```

5.3 Composing the DMMSG:

According to the analysis done in the UII module one can see that the system administrator doesn't follow the normal way of removing a routing node from a network. It may now be possible that there exists a so-called 'insider attack', that is, if someone operates outside user model thresholds, operations on sensitive resources, inappropriate creation and manipulation of resources.

This assumption is coded in the following DM-rule (written in pseudo-code):

```
IF (hypothesis-number = 600400)
THEN
      attack classes = 'insider attack'
      level_of_conf = unchanged
      level_penet = '2'
      params = [service; insider]
```

The level_of_conf remains unchanged because there is no additional knowledge (in the DM) available that there is really an 'insider attack'. It may well be that the administrator acts in a very urgent situation and it is not possible to follow the normal procedures

The `level_penet` which is an indication for the gravity of an intrusion is set to '2' (Table 4). The field `params` contains additional information for the SSO; 'service' means that a service (routing) is the target of the actual intrusion, 'insider' specifies the intruder which has in the actual case, a valid account for the network.

Level 0	INFORMATION, NOTIFICATION
Level 1	ATTEMPT of an intrusion which does not succeed
Level 2	ABNORMAL BEHAVIOUR of a user but no more information
Level 3	DAMAGE LIMITED to non-privileged users
Level 4	Intruder gets PRIVILEGED RIGHTS

Table. 4.: Instances of `level_penet`

To make sure that there is no misuse of hardware resources the SSO and the Countermeasure Module will be informed. The SSO can talk to the user (which has been identified via the UII-record) and can find out what's going on.

6. Summary and discussion

In this paper the design and development of a prototype of an expert system application for the detection of certain types of abnormal behaviour in open networks has been presented. Two modules of the system have been presented, the User Intention Identification (UII) module and the Decision Module (DM).

The User Intention Identification module is a module for the detection of anomalous behaviour by reasoning about the characterisation of the intentions of users. This module views the users of a system as using it in order to achieve certain goals by performing various tasks. This module aims to detect a range of malicious attacks i.e. in the cases where malicious tasks are composed of legal events. Since the basic actions for these tasks are allowable they cannot be detected by simple matching mechanisms. The examination of the whole rationality behind the execution of these basic actions has to be considered in relation to the general goals these actions are trying to fulfil (when composed to form tasks). Reasoning about the deviations observed in the execution of actions within a task in relation to the normal task execution (under the general goal-oriented constraints) offers an indication of the suspiciousness of the observed behaviour

The main task of the Decision Module on the other hand, is to produce a final decision about whether an attack has taken place or not. According to its architecture this module receives input from various analysis modules of the system and tries to combine this input in order to provide useful aid to the SSO. The objective is to aid the SSO determine the seriousness of the possible attack and also to help,him to look for the appropriate countermeasures. This is fairly straightforward when a known attack is the issue but in the case of unknown attacks the SSO has to consider a number of constraints.

One of the main problems during the design and development of the system was the communication problems between the two modules. These communication problems range from the actual implementation of the data exchange between the two modules to the determination of the limits where the behaviour analysis finishes and the decision making process starts.

The development of the current version of the modules in the frame-based expert system-shell CLIPS was proved to be very successful. The speed of the running system is fairly adequate and an evaluation process has been planned and is now being carried out.

7. References

[1] J. Darzentas and T. Spyrou. Functional Specifications of SECURENET Components, *In: SECURENET System Development Plan, CEC RACE Report, R2057.EXP.DR.L.050B1*, pp. 53 - 143, 1992.

[2] J. Giarratano and G. Riley. Expert Systems: Principles and Programming. PWS Publishing, Boston, MA., 2nd. edition, 1994.

[3] D. Karagiannis, C. Mayr, R. Telesko: Design of the Decision Module, Deliverable Nr. 17, Internal Publication of the RACE-Project R2113, February 1995.

[4] NASA Johnson Space Center, Houston, TX, "Clips Programmer's Guide, Version 6.0, JSC-25012", June 1993.

[5] SECURENET Project, Deliverable 3, Overall System Concept, Technical Report R2057.EXP.DR.L.030B1, RACE Programme, 1992

[6] SECURENET Project, Deliverable 5, System Development Plan, Technical Report R2057.EXP.DR.L.050B1, RACE Programme, 1992

[7] Spirakis P., Katsikas S., Gritzalis D., Allegre F., Darzentas J., Gigante C., Karagisnnis D., Kess P., Putkonen H. and Spyrou T., SECURENET: A Network Oriented Intrusion Prevention and Detection Intelligent System, *Network Security Journal*, vol. 1, no 1, Nov. 1994.

[8] Spirakis P., Katsikas S., Gritzalis D., Allegre F., Darzentas J., Gigante C., Karagisnnis D., Kess P., Putkonen H. and Spyrou T., SECURENET: A Network Oriented Intrusion Prevention and Detection Intelligent System, IFIP SEC93, *Proceedings of the 10th International Conference on Information Security*, May 1994.

[9] T. Spyrou, J. Darzentas. Specifications and Design of the Intention Identification Module, Deliverable Nr. 17, Internal Publication of the RACE-Project R2113, December 1994.

[10] E. Sutinen, "/DEL/IMP/4.2/OULU/EPS/050395; Implementation of SICS", *Securenet II deliverable*, 1995.

[11] E. Sutinen and H. Putkonen, "/DEL/SPC/4.1/OULU/EPS/050594; Specification of SICS", *Securenet II deliverable*, 1994.

[12] E. Sutinen and H. Putkonen, "/TRP/DES/4.2/OULU/EPS/050395; Design of SICS", *Securenet II deliverable*, 1995.