1. Title of the Project / Number of Annual Report

Project 1994P: Formal Methods for Information Protection Technology

Task 2. MATHEMATICAL FOUNDATIONS, ARCHITECTURE AND PRINCIPLES OF IMPLEMENTATION OF MULTI-AGENT LEARNING COMPONENTS FOR ATTACK DETECTION IN COMPUTER NETWORKS

Report #3

2. Contracting Institute

St. Petersburg for Informatics and Automation

3. Participating Institutes

None

4. Project Manager, phone number, fax number, e-mail address

Prof. Igor V. Kotenko, (812)-323-3570, (812)-328-0685, ivkote@iias.spb.su

5. Commencement Date, Duration

December 1, 2000, 36 months

6. Brief description of the work plan: objective, expected results, technical approach

Brief description of the work plan

B-1. Development of the learning task ontology, allocation of learning tasks over generic learning agents	1-3 Quarters
Interim Report #1 summarizing the results of the task B-1	3 Quarter
B-2. Development of architecture of the Multi-agent Learning System and mathematical methods realizing learning functionalities of the generic agents	4-6 Quarters
Submission a paper in an International Journal	5 Quarter
Interim Report #2 summarizing the results of the task B-2	6 Quarter
B-3. Development of the protocols of inter-level intelligent agent interaction (negotiation), generalization of the particular agent decisions according to the meta-classification approach and development of architecture of the Multi-agent Learning System as a whole	7-8 Quarters
B-4. Development of object-oriented conceptual project of the Multi-agent Learning System	6-8 Quarters
B-5. Development of the software prototype of the Multi-agent Learning System implementing theoretical results of research and its evaluation	9-11 Quarter
Interim Report #3 describing the results of the task B-4 and partially developed software prototype of the components of the Multi-agent Learning System	10 Quarter
B-6. Evaluation of the properties, advantages and disadvantages of the developed architecture and mathematical methods implemented within the prototype of the intelligent Multi-agent Learning system	12 Quarter
Final report, summarizing the results of simulation of the developed Multi-agent Learning System aimed at attack detection, as well as final conclusion concerning the research results on the task 2 as a whole	12 Quarter

Notice: The grey shaded rows correspond to the tasks to be solved during the third year research.

Objective

The objectives of the Task 2 of this project are development of mathematical foundation, agent-based architecture, and principles of implementation of the Intrusion Detection Learning Systems operating in a parallel with the Computer Network Assurance System.

Expected results

1. Learning task ontology.
2. Allocation of learning tasks over generic learning agents and development of the architecture of their interaction within Multi-agent Learning System.
3. Mathematical basis and algorithms realizing learning functionalities of the particular agents and software prototypes of the components of the Multi-agent Learning System based on theoretical results of the research.
4. Simulation-based evaluation of the properties, advantages and disadvantages of the developed multi-agent model and architecture of the Multi-agent Learning system aimed to support adaptability and learnability of the Network Security System.

Technical approach

A key issue of an intrusion detection learning task is selection and development (if necessary) accurate and efficient methods and algorithms specialized to the data structures specifying intrusions. This data specifying is represented mainly in the form of timely ordered sequences of audit data of various lengths specified in terms of repeatable symbols. These symbols correspond to the preprocessed messages of the input traffic at a host port of the computer network. In case of the distributed attack as well as in case of the normal distributed users' activity, a set of such sequences specifies a user activity. Mining of such data is a challenge in knowledge discovery from databases.

The approaches to be used within the last task are threefold. The first class of them is based on the specification of each input sequence (example) corresponding to a class of attacks or normal activity as a word of an unknown formal Context Free language. In this case the learning task may be reduced to a task of the inductive formal grammar recovery. The second class of approaches to be used is based on statistical properties of the examples specifying normal and abnormal activity of the user(s) accessing to network resources. The third class of methods intends to solve the task of the rule extraction from the training data sample and its result is specified in terms of predicates given over higher level concepts like patterns.

An important issue is a decomposition of the whole intrusion detection learning task into multitude of sub-tasks according to the ontology of attacks, and their allocation among specialized learning software agents. Intrusion detection learning system will be implemented on the basis of multi-agent architecture. Within it, each specialized learning agent is considered as the generic one which is capable to extract patterns of a particular class (association rules, frequent patterns, production rules given over patterns, etc.), and it should be able to extract knowledge from data of the fixed format (event sequences, sets of patterns, subset of rules, etc.).

For the purpose of the formal specification of learning agent interactions, which are especially necessary in learning detection of distributed attacks, the idea of meta-classification will be used. This idea entails the necessity of distributed learning which corresponds to the multi-level learning aiming at fusion knowledge resulted from local learning procedures realized by particular learning agents.

7. Technical progress during the second year (for 3nd annual reports)

Technical progress during the second year is fully compliant with regard both to the tasks predefined the Work plan and to the schedule of their completion.

Achievements of the second year

The basic achievements of the second year correspond to the tasks scheduled. These tasks and as follows:

1. Development of architecture of the Multi-agent Learning System and mathematical methods realizing learning functionalities of the generic agents.

2. Development of the protocols of inter-level intelligent agent interaction (negotiation), generalization of the particular agent decisions according to the meta-classification approach and development of architecture of the Multi-agent Learning System as a whole.

3. Development of object-oriented conceptual project of the Multi-agent Learning System.

The main results obtained during first year research are as follows.

1. Analysis and development of the formal model and architecture of the particular generic agents of the multi-agent learning system.

The following two tasks were the subjects of activity.

The first task concerns with the development of the mathematical and algorithmic basis supporting intrusion detection learning. Within this task the main efforts were focused on the study of the existing methods and algorithms, which have been published recently and which are used at present by the specialists in the area of interest. Also, the subjects of study were other methods and algorithms developed within knowledge discovery from databases area, which can potentially be used fro intrusion detection learning. In this study, the main attention was paid to the methods of frequent pattern mining, in particular, mining frequent patterns from temporal sequences. At the same time, several methods were implemented with the purpose of preliminary evaluation of their properties and usefulness in the learning task of interest. In particular, such kind of the software is developed for *FP-growth* algorithm aiming at mining frequent patterns and association rules from transactional databases. Also the adaptation of the methods and algorithms developed by the authors of this research and analysis of their capabilities within intrusion detection learning task was carried out. In particular, it was developed experimental software for *GK2* method for mining rules from relational databases. This software together with the Visual Analytical Mining (*VAM*) method also developed by authors was tested on the basis of the intrusion detection training data that was used in KDD Cup-99 competition of data mining and knowledge discovery programs. The exploration of these methods is carried out within meta-classification learning task that aims at learning fusion of the local decisions produced on the basis of local sources of data resulting from monitoring security aspects of a computer network.

The second task aims at development of formal models and architectures of the particular classes of agent forming multi-agent learning system under development. Such models and architectures were developed for the following classes of agents: Class of learning data management agents; Class of classifier testing agents; Class of meta-data forming agents; and Class of learning agents which, in turn, comprises the following sub-classes: (i) Class of *agents* designed for the learning classifiers of attacks, which input data are represented as temporally ordered sequences of events, and (ii) Class of *agents* designed for the learning of classifiers that extract knowledge from learning data represented in the form of attribute vector.

Each of these classes comprises reusable and particular components. The standard components are responsible for receiving syntactic analysis and sending of the messages, with which the agents exchange, and also the standard parts of the semantic mechanisms of the input and output messages processing that is implemented on the basis of *state machine* framework (according to the *UML* language terminology). "Individuality" of each agent class is reflected by specifications of the particular state machines (alphabets of inputs, states and transitions, state transition functions and corresponding state machine actions specified in terms of the behavior scenarios and something else), and also by concrete contents of data- and knowledge bases. The formal models of all standard components of agent classes are developed. Some of them are tested as software components. Also some models of the special components of the agent formal models and architectures are developed. All the above models are specified formally in terms of *USE CASE DIAGRAM*, which corresponds to a visual representation of the components functional behavior specified formally in terms of the *UML* formal language.

2. Analysis of computer network intrusion detection learning (IDL) task and determination of specific problems of IDL.

Intrusion detection learning (IDL) task differs from typical data mining and knowledge discovery one in many respects. The main peculiarities leading to several specific problems of IDL technology result from the peculiarities of learning data.

Analysis proved that these peculiarities result from *distributed* nature and *heterogeneity* of data. The traces of illegitimate activity of users (erroneous commands of users and attacks against

computer and its information resources) are reflected in multiple distributed and heterogeneous data sources (*tcpdump*, OS system calls audit trail, application audit data, etc.). The data can be represented in different data structures (relational, sequential, temporal sequential) and measured in different measurement scales (Boolean, categorical, linear ordered, real), be of different accuracy and reliability, they may be incomplete and uncertain, contain missing values, etc. These properties along with other ones put specific problems within IDL system design and implementation.

The most important and difficult problems are as follows: (1) the necessity to provide monosemantic understanding of the terminology used by different components of IDL system, (2) entity identification problem, (3) problem of diversity of data measurement scales of training data components. These problems and some others form together so-called "data non-congruency problem". The problems are analyzed and approaches capable to cope with these problems within IDL task are proposed.

The basis for solving the *first problem* is "ontologism", i.e. ontology-centered approach. The technology for development and implementation of application ontology is proposed. This technology supports for the development of shared (common for all software components) and private (held only by particular agents of IDL system) components of application ontology that are "coherent" with the intrusion detection problem ontology. In addition, the ontology-centered approach makes it possible to resolve several other problems.

The *second problem* corresponds to a so-called *entity identification problem*. Each local data source specifies an entity (object to be classified) only partially. Its complete specification is made up of data fragments distributed over the data sources. Therefore, a mechanism to identify such fragments is needed to make it possible to retrieve, collect and analyze together distributed data about the same user activity. It is noteworthy to notice that some fragments of data associated with the above entity can be absent in some sources. Within this project, the *entity identification problem* is solved in the following manner. In the application ontology, for each entity, the notion of entity identifier ("*ID entity"*) is introduced. This entity identifier is a primary key of that entity (in analogy with the primary key of a table). For each above identifier, a rule within the framework of the application ontology is defined, which can be used to calculate the key value. For example, a unique combination of a set of this entity attributes can be one such rule. A rule is defined at the level of each local data source, and this rule must uniquely connect the entity identifier and the local primary key in this source. This rule specifies: how to derive the local primary key (from the value of which the values of all the entity attributes in this local source can be further derived) from the entity identifier value; how to derive the entity identifier value from the value of the local primary key of the source.

Next, the data specifying the same entity can be represented in different sources in terms of different data structures (images, signals, expert statements assigned a measure of uncertainty, scalar data presented in Boolean, categorical, ordered and/or numerical measurement scales, etc.).

And *the last* but not the least problem is that the sets of attributes of different data sources may be overlapping. At that the *same* or "*similar*" properties of the entity can be presented in different data sources in different ways, for example, in different measurement scales, with different accuracy, and so on. This problem is solved by the appropriate strategy of fusion of data of different sources that is known as fusion decisions produced on the basis of local data sources.

The respective approaches are developed with regard to all above-mentioned problems. In the multi-agent architecture of IDL system the "data non-congruency problem" is solved due to the making use of special agents that are *Data source managing agents* (associated with the particular data sources) and *KDD master agent*, which is a component of meta-level part of multi-agent system. The idea of using of such kinds of agents in IDL system architecture is new and seems promising.

3. Analysis of training and testing data for intrusion detection learning.

The analysis of intrusion detection learning task permitted to determine how distributed and heterogeneous data must be processed jointly and which requirements must be met by methods of data mining and KDD from architectural point of view. The mathematical methods of data mining and KDD themselves that cover the needs of IDL task can further be determined on the basis of analysis of structures of training and testing data peculiar to available data sources. The results of this analysis are shortly presented below.

The training and testing data for the IDL system exists in many forms and can be received from different data sources. *The taxonomies of these data sources* can be formed by different tags: (1) location of source and software generating data; (2) processing level; and (3) an object, with which the data are associated:

- *The taxonomy, which classifies data sources due to location of source and software generating data*, includes two main sources: network-based, and host-based. The network-based sources depend on network layers and used protocols. Host-based sources are represented by operating system audit trail, system logs, and application-related audit data.
- *The taxonomy, which classifies data sources due to processing level*, consists from primary (raw), preprocessed, and generalized sources. *Primary* sources are network traffic, host command (system calls) traffic, and data from other sources. *Preprocessed* sources are *tcpdump* (for packets), preprocessed OS audit trail, system logs, and audit data of different applications. *Generalized* sources are generated by statistical processing of preprocessed sources.
- *The taxonomy, which classifies data sources due to an object, with which the data are associated*, comprises network-based sources (packets, connections, all network traffic) and host-based sources (traffic within a connection, processes, users, files and directories, disks, system registry, etc.).

The data sources that are supposed to use in IDL case study are as follows:

- Network-based sources: Preprocessed *tcpdump* data for IP, TCP, UDP, ICMP packets, and *Tcpdump* statistical data.
- Host-based sources: Preprocessed OS audit trail and statistical OS audit data; System logs (for example, log and statistical data of commands run by users plus resource, log and statistical data of all login failures, log and statistical data of all user logins/logouts and system startups and shutdowns); Application audit data (for example, FTP logs and FTP statistical data, TELNET logs and TELNET statistical data, mail logs and Mail statistical data, HTTP logs and HTTP statistical data, DNS logs and DNS statistical data).

Four *typical structures* of data that can be used in IDL task: Time-based sequential data, Sequential (ordered) data, Relational (non-sequential) data, and Transactional data.

The *typical measurement scales* of ID learning data are as follows: Binary (or Boolean), Categorical, Linear ordered, and Real.

4. Development of mathematical methods realizing learning functionalities of the generic agents.

The multitude of methods that covers the needs of this task includes:

(1) methods for combining decisions produced by base-level classifiers on the basis of different data sources containing fragments of information about status of host operation security, and

(2) data mining and knowledge discovery techniques used for training and testing of base-level classifiers.

Two of *methods used for combining decisions* are chosen and described in detail. They are meta-classification method and competence-based method with some modifications developed by authors of this Project. At that the first method was also validated on the basis of KDD Cup-99 case study.

From the many existing methods *of data mining and knowledge discovery* developed for different types of data structure three basic methods were selected. The selection is based on analysis of data structures that can be perceived or computed on a host with the purpose of analysis of this host security status. The selected methods are:

1. **FP-growth** (*Frequent pattern growth*) method of frequent patterns and association rules mining aiming at extraction of useful patterns from transactional (sequential) data. This method was proposed recently and outperforms all known methods of the same type. This conclusion is made on the basis of theoretical analysis of its complexity and also on the basis of its software implementation carried out by participants of this Project. Particularly, this method was implemented as a component of the *Server of learning methods*. Experiments carried out proved high efficiency of the *FP-growth* approach and which was the reason to come to the conclusion that it is reasonable to use this method instead of widely used group of methods based on *Apriori* algorithm.
2. **VAM** (*Visual Analytical Mining*) aiming at mining rules and other kinds of pattern from numerical data. This method was developed in depth and validated over several data sets

from UCI repository. It is already implemented as a component of *Server of learning methods*.

3. **GK2** algorithm aiming at mining discrete data that was proposed, developed, implemented and validated by authors of this Report. The method is theoretically sound and showed good efficiency. The advantage of this method is that it can also be used for extraction rules from data with missing values and at that it does not require to use a prognosis of missed values.

Case study "KDDCup-99" was used for the validation of both latter algorithms, and also multi-agent architecture of IDL system and implementation technology.

5. Development of the protocols of inter-level intelligent agent interaction (negotiation).

The following types of *the protocols of inter-level intelligent agent interaction (negotiation)* were developed: (1) protocols for operation with particular data sources; (2) protocols for creation of global coherent problem ontology, shared and private components of application ontology; (3) protocols for combining decisions of source-based classifiers.

The most complex protocols are the ones for the task of creation of global coherent problem ontology, shared and private components of application ontology. This task consists in creation and synchronization of the source-based fragments of application ontology and their synchronization with the Data Fusion (DF) problem ontology.

These protocols are served for interaction of the IDLS entities located on different hosts during the design of the draft version of the application ontology and its iterative modification in the process of providing coherency. We have determined the protocols for creating the initial (basic) version of the application ontology (we called it the meta-protocols) and the protocols for further synchronization of the ontology during its iterative coordination with the local components of the ontology, as well as during any modifications to it.

We considered two meta-protocols: “top-down” and “bottom-up”.

In the first case the expert of the meta-level responsible for forming the global ontology forms its basic variant that includes the list of basic application entities with the minimally necessary set of attributes and specifies the identifiers of entities. In the case of using multi-agent architecture of IDLS, special agent ("KDD master") managed by the meta-level expert sends out the basic variant of local fragments of the application ontology to the respective agents situated on the local data sources ("Data source managing agents" – DMAs) for analysis, correction, further expansion and filling. The DMAs of local sources managed by experts conduct modifications and extensions of the received ontology version aiming at whole ontology coherency providing. The synchronization of changes and extensions of the first and next versions of ontology made by agents of the data sources is conducted by the meta-level agent step by step via exchange of messages with data source agents. The contents of the synchronization protocol consists in multi-phase negotiations, at that each source agent negotiates only regarding the shared and its own private part of the application ontology. These negotiations are carried out with the KDD master and results in the development of the application ontology which is coherent with problem ontology and has no contradictions on the level of application. All these procedures are performed under the supervision and with the active participation of meta-level expert and local source experts, interacting through their agents. After processing the above information, the local source DMA prepares their proposals as to modifying and/or expanding the local components of the domain ontology and sends these proposals to the KDD master.

In the “bottom-up” protocol, local source experts first form the basic variants of the application ontology with regard to its shared part and their own private one, and then KDD master under supervision of meta-level expert performs the merging, coordination and correction of the received components of the application ontology to prepare the next basic variant of it. After that, the corresponding parts of this variant are sent to the local sources DMAs for further corrections if necessary. The subsequent work is performed in the manner similar to the above step of protocol.

In both protocols, the central component is its part that deals with the synchronization of the application ontology components proposed by KDD master and DMAs of the local sources of the application ontology.

In considering the interactions between the IDLS components we have to take into account the possible spatial distribution of data sources and a possibility of non-reliable communication channels between the data sources and the meta-level host-server. For the realization of interaction mechanisms working under such conditions, protocols utilizing the two-phase lazy transactions

have been used. These protocols are essentially similar to the database synchronization protocols, except that in used synchronization protocols, the verification of modifications carried out on the meta-level server is impossible. The overriding decision-making right in the area of modifications to the ontology resides with the meta-level application expert who bear the main responsibility for forming and supporting the global application ontology. Their duties also include periodic review and verification of the modifications to the ontology proposed by the application experts who work with the local data sources.

6. Development of the technique for generalization of the particular agent decisions according to the meta-classification approach.

The developed technique for generalization of the particular agent decisions is based on a specified hierarchy of interaction of particular classifiers in process of producing a global decision on the basis of hierarchical combining of decisions of lower level classifiers.

Several basic approaches to combining decisions of multiple base-level classifiers have been analyzed. They can be grouped into four groups: (1) Voting algorithms; (2) Probability-based or fuzzy algorithms; (3) Meta-learning (meta-classification) algorithms based on stacked generalization idea; (4) Meta-learning algorithms based on classifiers' competence evaluation.

The meta-classification and competence-based methods were used and adapted in the Intrusion Detection Learning System (IDLS) under development. It is noteworthy to mention that both above methods cannot be used in "straightforward" manner because of peculiarities of data. Thus these methods were adapted for application of distributed learning and decision making procedures. Again, most of these peculiarities are of technological kind.

7. Development of architecture of the Multi-agent Learning System.

The developed architecture of the Intrusion Detection Learning System consists of the local data source components and the meta-level components.

The following main agents were included in IDLS:

- *Intrusion detection KDD master agent*, which realize distributed intrusion detection learning application ontology design, design of meta-model of decision making on intrusion detection, and distributed learning management.
- *Meta-level intrusion detection KDD agent*, which is for distributed learning management, design of meta-model of decision making on intrusion detection, sending of the decision making structures to local level agents;
- *Intrusion detection agent-classifier of meta-level* realizing distributed learning management and decision making on intrusion detection.
- *Data fusion management agent*, which is intended for design of meta-model of decision making on intrusion detection, distributed learning management and decision making.
- *Intrusion detection KDD agent of a source*, fulfilling design of meta-model of decision making and distributed learning management.
- *Intrusion detection agent-classifier of data source*, executing distributed learning management and decision making.
- *Data source management agent*, which is for decision making on intrusion detection, distributed application ontology design, design of meta-model of decision making, distributed learning management and data source monitoring to detect receipt of new data.

8. Development of object-oriented conceptual project of the Multi-agent Learning System.

Object-oriented project of IDLS was developed in terms of *Uses cases* diagrams, *collaboration* diagrams, *state-chart* diagrams and *component* diagrams.

Object-oriented project of IDLS includes the following specifications: the high level behavior of IDLS; the Base Classifier's decision making on intrusion detection; the Meta-classifier's decision making on intrusion detection; the IDLS agents' behavior in the processes of data preparing, informative features search, getting meta-properties of data, classifier learning, Meta-classifier learning, measurement scale transformation, construction of basic model of decision making knowledge base, basic classifier validation procedure, firing particular rule of Base Classifier knowledge base, design of meta-model of decision making, distributed application ontology design; etc. On the whole, they provide the necessary specifications of components of the system needed to write software code.

8. Technical progress during the year of reference

Technical progress during the year of reference is fully compliant with regard both to the tasks predefined the Work plan and to the schedule of their completion.

Achievements of the year of reference

The basic achievements of the year of reference correspond to the tasks scheduled. These tasks and as follows:

1. Development of the software prototype of the Multi-agent Learning System implementing theoretical results of research and its evaluation.

2. Evaluation of the properties, advantages and disadvantages of the developed architecture and mathematical methods implemented within the prototype of the intelligent Multi-agent Learning system.

The *main results* obtained during the year of reference are as follows.

1. *Development of the software prototype of the Multi-agent Learning System implementing theoretical results of research and its evaluation.*

For implementation of the Multi-agent Intrusion Detection Learning System (MIDLS) prototype, the technology and software tool called Multi-agent System Development Kit (MAS DK) is being elaborated and used. This tool is implemented on the basis of Visual C++ 6.0, JAVA1.3 and XML programming languages.

The MIDLS prototype architecture consists of the local data source components and the meta-level components.

The complete list of the IDLS software classes and procedures that were developed in the Project is given in the table 1. It should be noted that architecture of IDLS designed by making use of MASDK implements agents behavior (its functionalities) in terms of state machines. Decomposition of the entire IDLS functionalities and composing the subtasks to be allocated to the particular agents is made in implementation-oriented mode. This is the reason why the functions of agents are named in table 1 as state machines of the respective destinations. Such a terminology is more understandable for designers of IDLS that use MASDK as a design technology support software tool.

Agents marked by symbol "*" are not the subject of the research in this project. These agents are the components of Intrusion Detection System designed by IDLS. Nevertheless, the simplified versions of such agents are also under development due to the necessity to validate the developed technology of IDLS design.

Table 1. List of functions, libraries or modules IDLS

Agent name	Names of function, library or module
KDD Master	*Editor of meta-level ontology*
	State machines providing interaction with editor of meta-level ontology
	Editor of decision making meta-model
	State machines providing interaction with Editor of information fusion meta-model
	State machine querying characteristic of data sources
	Information Fusion meta–model editor
	State machines providing interaction with Information fusion meta–model editor
	State machine responsible for forwarding learning tasks to KDD agents
	State machine responsible for forwarding Decision making meta-model to the decision making agent
	State machine responsible for forwarding training and testing data sample specification
	State machine responsible for preparation of meta-data used for meta-classifier training and testing
	Function querying generalized specification of data sample

Agent name	Names of function, library or module
	Function querying identifiers
Meta-level KDD agent	*Interface of the meta-learning program tracing (debugger of meta-learning)*
	State machine implementing interaction with meta-learning program tracing
	State machine receiving information about finalizing of the base classifier learning
	State machine receiving meta-learning task specifications
KDD Agent (of a source)	*State machine receiving local learning task specification*
	Basic state machine of user interface supporting training and testing
	Interface of the managers of classifiers' status
	State machine responsible for resending classifiers' attributes
	Interface for estimation of the coverage factor of the rules
	State machine managing rule extraction procedure
	Interface supporting the classifier attribute tuning
	State machine supporting the classifier attribute tuning
	Interface supporting transformation of the data measurement scales
Server of learning methods	*Data mining function "vam"*
	Data mining function "gk2"
	Data mining function "FP-grows"
	Data mining function "Temporal mining"
** Source-based classification agent (base classifier) (BC)*	*State machine receiving rules generated and classification attributes*
	Decision making state machine of classifier
	State machine informing about readiness of a base classifier to produce decision
	Function responsible for monitoring of arrival of input data
	Decision making based on particular rules
	State machine receiving attributes specifying a classifier
** Agent-classifier of meta-level (meta classifier) (MC)*	*Decision making state machine of Meta- classifier*
	State machine receiving decision making meta-model
	State machine receiving attributes specifying meta-classifier
Information Fusion (Decision combining) management agent	*Interface of the decision combining support system*
	State machines of the decision combining support system
	Decision combining system
	State machine receiving input data arrived
Data source managing (DSM) agent	*State machine performing data preparation*
	Function responsible for data extraction and transformation
	State machine receiving specification of notions
	State machine receiving attributes of data
	Interface for tuning of the application ontology notion interpretation
	State machine implementing interface for tuning of the application ontology notion interpretation
	State machine receiving generalized data properties
	State machine performing receiving and forwarding of the identifier's list
	State machine preparing a data sample
	Function responsible for preparing of a data sample

Agent name	Names of function, library or module
	State machine performing monitoring of the data source
	Function responsible for monitoring of the data source
	State machine responsible for receiving attributes of classes

According to the technology implemented in order to build IDS and IDLS as applied Information Fusion MAS, they are firstly specified in the *System Kernel* of the MASDK by making use of Generic agent as a template for specification of agent classes. In parallel problem domain ontology of IDL is also specified. Next step consists in specification of the IDS and IDLS agent classes and the shared component of application ontology. Then agent classes are replicated into agent classes' instances and installed in predefined computers. The resulting IDLS has to be further "filled in" by particular content (data and knowledge interpreting particular ontology notions, providing particular agent procedures with concrete data). After that the IDLS operates in the environment independently of MASDK. In the learning mode, training and testing of IDS classification and decision combining agents is fulfilled.

In accordance with the given data sources, the configuration of the agents' instances of the IDLS and IDS software prototype is formed in the following manner: (1) For each data source, one logical host and three instances (according to the number of data sources) of software agents *DSM, BC, KDD agent* are specified; (2) To specify the meta–level component of IDLS and IDS, one or several logical hosts are specified. In each logical host, one instance of software agents *MC* and *KDDl* is specified; (3) Agents of class *KDD Master* that support the management of training and decision making processes are located on the same logical host.

The *base classifier training* scenario by the *KDD–agent of a source* consists of a number of particular subtasks performed in a certain order: Scales conversion to the scales, for which implemented algorithms exist (if the training data contains the attributes of the ordinal or categorical type); Search for rules in favor of class; Tuning of the decision-making mechanism; Testing of classifier; Sending classifier's description to the agent of base classifier.

The *training of meta classifiers* is based on usage of data computed by the base classifiers which decisions are combined by the respective meta classifier. The computation of input data for training and testing of meta classifiers is the function of the *Meta level KDD agent*.

The main task of the *data source management agent* is enabling direct access to data and subsequent transformation of the data to the format of the shared application ontology.

2. Evaluation of the properties, advantages and disadvantages of the developed architecture and mathematical methods implemented within the prototype of the intelligent Multi-agent Learning system.

Design and implementation of the components of the software prototype of Intrusion Detection Learning System (IDLS) was accompanied by demonstration of the practical use of the developed methodology, technology and supporting software tool thus making it possible to validate them.

For experiments we determined *the categories and instances of attacks to be used in the case study*. To generate training and testing data we selected *four types of attack categories:* Probing; Remote to local (R2L); Denial of service (DOS); User to root (U2R). The exemplars of attacks selected for case study are SYN-scan, FTP-crack attack, SYN flood, and PipeUpAdmin.

The data sources used and *generic data structures* representing training and testing data of the selected sources are described. We have chosen three *data sources for training and testing data*: network-based (traffic level), host-based (operating system level) and application-based (FTP-server level). Each data source is represented by four *generic data structures*. These data structures correspond to the data produced on the basis of raw data processing. These data structure are as follows:

1. Time ordered sequence of values of binary vectors of parameters specifying significant events of a level (traffic level, OS logs and FTP-server logs);
2. Statistical attributes of particular connections (performance of a user) manifested in a data source (traffic level, OS logs and FTP-server logs);

3. Statistical attributes of traffic (users' activity) during the short term time intervals;
4. Statistical attributes of traffic (users' activity) during the long term time intervals.

The *instances of data structures representing training and testing data sets* to be used in data mining and KDD procedures are specified. The data structures of the network-based source (traffic level) are produced on the basis of processing of *tcpdump/windump* data. The data structures of host-based source (operating system level) were produced on the basis of processing of operating system log *Security* (for Windows 2000/XP). The data structures of application-based source (FTP-server level) were produced on the basis of processing of FTP-server log. We used for generating these data structures *TCPtrace* utility and several programs developed by the authors of the Project.

The *examples of training and testing data instances* of each selected data source as they are used in learning procedures in case study were represented. These examples were presented for all exemplars of attacks concerning to four attacks categories selected. We described also examples of data fixed for normal users' activity.

In experiments with the software prototype of Intrusion Detection Learning Multi-Agent System (IDL MAS), including components of IDLS and Intrusion Detection System (IDS), the following meta-classification and data fusion model was realized:

(1) A set of base classifiers was used which output was represented as a stream of decisions appeared at the ordered by random time moments;

(2) If a new event occurred in a base classifier then this event was transmitted to meta-level and all other base classifiers submitted to meta-level its latest decisions of their formed decision streams;

(3) Each event of the output stream of each particular base classifier was assigned some "life time" mark, and if this event was not used within the respective time interval then it was not taken into account.

The analysis of the testing results of the IDL MAS software prototype allowed to conclude that the agent-based approach to IDL and elaborated methodology, technology and software tool constitute a promising starting platform for further research and development of the prospective IDLS.

The developed software prototype of IDLS showed, for instance, the following results:

(1) The probability of perfect classification on testing dataset of size of 789 on the basis of network layer data for Abnormal recognition is equal to 0,98;

(2) The probability of perfect classification on testing dataset of size of 138 on the basis of OS and application layers for FTPCrack attack recognition is equal to 1,00.

9. Current technical status

The progress in research fully matches the Work program and does not need in a refinement.

10. Cooperation with foreign partners

According to the Work plan, the *Interim Report* #3 was submitted to the Partner (to June 1, 2003) and the *Final Report* was submitted to the Partner (to December 1, 2003). These Reports contain the results of all predefined research.

In the *Interim Report #3*, the research results are presented in four chapters. Chapter 1 specifies the case study used for development of object-oriented project and software prototype of the components of IDLS. Chapter 2 presents conceptual description of the developed multi-agent technology for learning of intrusion detection and specifies high level protocol of interaction of IDLS components and user within the above engineering technology. Chapter 3 describes conceptually the developed model of "generic Agent" that is considered as a basic component supporting technology for applied software agent design and implementation. Chapter 4 presents the components of the object-oriented project of IDLS and describes the status of the IDLS software code development. Conclusion summarizes the main results of the second phase research presented in this Report.

In the *Final Report*, the results of the Project were represented in five chapters and two appendixes.

Chapter 1 introduces into the problems solved in the Project and outlines peculiarities of the intrusion detection learning task as well as the basic ideas that form the milestones of the Project

research. It outlines a brief overview of the modern days research on this subject. It also introduces generally the approach to computer network intrusion detection learning accepted in this Project. The Chapter presents the main concepts of logging and auditing of the events, happening in computer networks to be defended, analysis of the data structures and examples of audit data used in modern operating systems and applications. Besides it reviews the proposed decisions regarding to the structure of the learning data and their usage in attack detection and attack detection learning. Also the Chapter considers the basic aspects of the methodology of multi-agent intrusion detection learning adopted within this Project. The Chapter specifies mathematical methods of data mining and knowledge discovery that are available for use in the developed prototype of IDLS.

The *Chapter 2* presents conceptual description of the developed and implemented technology of multi-agent information fusion system engineering intended for design, implementation and deployment of applied multi-agent data and information fusion systems. The Chapter also describes the developed ontology specifying a high-level representation of the basic notions of Intrusion Detection Learning domain. The specific of the subject domain under research is that it combines knowledge and therefore, combines ontologies, from different subdomains, namely, "Data Fusion and Data Fusion Learning problem domain ontology", "Intrusion Detection application ontology" and "Intrusion Detection Learning application ontology". These ontologies are considered in this Chapter.

The *Chapter 3* is devoted to the conceptual view of the architectural and technological issues of IDLS design and implementation. The Chapter presents the developed multi-agent architectures of Multi-agent IDS and IDLS. The conceptual view of the structure of agent communication is outlined. The standard scenario of the IDS operation is outlined. Intrusion Detection Learning Scenario is suggested. It includes engineering of the shared component of the application ontology, design of the binary classification tree, design of the meta-model of decision combining (decision tree), engineering of base classifiers and meta–classifiers, testing of IDS as a whole and monitoring of the learning process. These phases of intrusion detection learning scenario are described.

The *Chapter 4* specifies the developed case study that was used for design and implementation of the software prototype of the components of IDLS. It determines the categories and instances of attacks used in the case study, data sources and data structures representing data of the selected sources, the instances of data structures, and the examples of training and testing data.

The objective the *Chapter 5* is to present the implemented components of IDLS and simulation results that in more details are given in Appendixes. *Appendixes* demonstrate the developed multi-agent technology destined for distributed intrusion detection learning and decision making on intrusion detection. These appendixes give the detailed description of the training and testing procedures, intermediate and final results in all the steps of the IDL including testing of particular base classifiers and also meta-classifier which produces the final decisions.

11. Problems encountered and suggestions to remedy

None

12. Perspectives of future developments of the research/technology developed

Proposal for continuation of the research supposed by this Project was submitted to Partner in September 2002.

Attachment 1: Illustrations attached to the main text

None

Attachment 2: Other Information, supplements to the main text

Brief content of the Reports submitted to the Partner

Interim Report #3

Final Report

Attachment 3: Abstracts of papers and reports published during the year of reference

1. V.Gorodetsky, O.Karsaeyv, and V.Samoilov. Distributed Learning of Information Fusion: A Multi-agent Approach. Proceedings of the International Conference "Fusion 03", Cairns, Australia, July 2003.

   Abstract. An important task of information fusion scope is learning of decision making and decision combining. This task that falls under Distributed Learning scope is a subject of the paper. It is supposed that distributed learning is carried out by a component of information fusion system performing supervised off-line training and testing of its decision making component. The core problem of a distributed learning component design does not concern particular data mining techniques. Instead of this, its core problem is development of an infrastructure and protocols supporting coherent collaborative operations of distributed software components (agents) responsible for distributed learning. The paper is focused on architecture of multi-agent information fusion systems possessing learning capabilities, on a technology supported by a software tool and on protocols of software tool agents' interaction, particularly, distributed data mining protocol. Solutions concerning the aforementioned aspects form the basis for the multi-agent information fusion system technology and respective software tool.

2. V.Gorodetsky, O.Karsaeyv, and V.Samoilov. Multi-agent Technology for Distributed Data Mining and Classification. Proceedings of the IEEE Conference Intelligent Agent Technology (IAT03), Halifax, Canada, October 2003.

   Abstract. The paper considers multi-agent distributed data mining technology. It is assumed that distributed data mining functionality can be implemented either within a separate system that should play the role of a software tool for multi-agent distributed classification system technology support or it can be embedded in applied multi-agent classification system thus providing it with off-line learning capability. Experience proved that the core problem of agent-based distributed data mining and distributed classification

does not concern particular data mining techniques although the latter problem is now paid the most attention. Instead of this, its core problem concerns interaction protocols supporting coherent collaborative work of distributed software agents responsible for the design of a classification system, in particular, for the design of its components responsible for distributed data mining. The paper considers architecture of multi-agent software tool focused on distributed data mining and proposes protocols for interactions of the software tool agents in the process of distributed design of an applied classification system. Proposed solutions form the basis for the multi-agent distributed data mining technology and respective software tool.

3. V.Gorodetsky, O.Karsaeyv, and V.Samoilov. Software Tool for Agent-Based Distributed Data Mining. Proceedings of the IEEE Conference Knowledge Intensive Multi-agent Systems (KIMAS 03), Boston, USA, October 2003.

   Abstract. The paper considers a software tool for multi-agent distributed data mining technology support and its usage for prototyping several applications from data and information fusion scope.

4. V.Gorodetsky, O.Karsaeyv, and V.Samoilov. Multi-agent Data and Information Fusion: Architecture, Methodology, Technology and Software Tool. Accepted for publication in the book "Data Fusion for Situation Monitoring, Incident Detection, Alert and Response Monitoring" E.Shakhbasyn and P.Vallin (Editors). To be published in Kluwer Academic Publishers. 2003.

   Abstract. The paper introduces the present-day understanding and statement of the data and information fusion problem and proposes a methodology, technology and software tool kit destined for the design, implementation and deployment of applied multi-agent applications pertained to this problem domain. The distinctive feature of the developed technology supported by software tool kit is that it is distributed and agent-mediated, i.e. it assumes a distributed fashion of designers' engineering activity mediated by agents performing the most part of routine engineering work and also providing coordination of designers' activity according to a number of protocols. The above technology and software tool kit supporting it are completely implemented and validated by prototyping several applications from data and information fusion scope.

5. V.Gorodetsky, I.Kotenko, and O.Karsaev. Multi-agent Technologies for Computer Network Security: Attack Simulation, Intrusion Detection and Intrusion Detection Learning. International Journal of Computer Systems Science and Engineering. vol.18, No.4, July 2003.

   Abstract. The paper presents an experience in applying multi-agent technology to the design and implementation of multi-agent systems (MASs) intended to cooperatively solve the current critical tasks in the area of computer network security. The MASs in question are Agent-Based Simulator of Attacks against Computer Networks, Multi-Agent Intrusion Detection System and Multi-Agent Intrusion Detection Learning System. Each of these MASs is based on strict formal frameworks proposed by authors and designed and implemented as software prototypes on the basis of common agent technology supported by Multi-Agent System Development Kit developed with the authors' participation. The paper outlines the above-mentioned MASs and analyses the advantages of applying multi-agent technologies to computer network security problems.

6. V.Gorodetsky, O.Karsaev, I.Kotenko, V.Samoilov, M.Stepashkin. Multi-agent system of intrusion detection learning. Proceedings of III Inter-regional Conference "Information Security of Russia Regions", vol.1. St. Petersburg, November 25-27, 2003. (in Russian).

   Abstract. The paper considers architecture and particular components of Multi-agent Intrusion Detection Learning System (MIDLS), providing the adaptability property of Multi-agent Intrusion Detection System (MIDS) to new attacks. The protocols of MIDLS operation, the structure types and the exemplars of training and testing data are represented. The experiments with MIDLS and MIDS prototypes are described.