Annual Technical Report

For the second year
(December 01, 2001 - November 30, 2002)

1. Title of project

Formal Methods for Information Protection Technology

2. Contracting Institute

St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences (SPIIRAS)

3. Participating Institutes

None

4. Project Manager

Dr. Oleg V. Karsayev
(812)-323-3570,
(812)-328-0685,
ok@mail.iias.spb.su

5. Commencement Date, Duration

December 01, 2000, for 3 years

6. Partner

European Office of Aerospace Research and Development

SPIIRAS Director,

Prof. Rafael M.Yusupov

Project Manager

Dr. Oleg V. Karsayev

December 2002

Task #1

1.1. Title of the Project / Number of Annual Report

Project 1994P: "Formal Methods for Information Protection Technology"

Task #1. "Formal Grammar-Based Framework, Model and Software Tool Prototype for Simulation of Distributed Attacks on Computer network"

Annual Report #2

1.2. Contracting Institute

St. Petersburg for Informatics and Automation of the Russian Academy of Sciences

1.3. Participating Institutes

None

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

Dr.Oleg V. Karsaev , tel: +7(812)-323-3570, Fax: +7(812)-328-0685, e-mail ok@iias.spb.su

1.5. Commencement Date, Duration

December 1, 2000, 27 months

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

Brief description of the work plan

A-1. Development of the specification of the representative set of distributed attacks defined on the macro-level.	1-3 Quarters
A-2. Development and selection of mathematical methods and techniques realizing the attack modeling.	2-3 Quarters
Interim Report #1 summarizing the efforts of the tasks A.1.	End of 2 Quarters
A-3. Development of the object-oriented project of the Attack Simulator–software tool prototype for simulation of attacks on the computer network	4-6 Quarter
Interim Report #2, summarizing the efforts of the tasks A.1 and A-2.	End of 4 Quarter
A-3. Development of the object-oriented project of the Attack Simulator–software tool prototype for simulation of attacks on the computer network	4-6 Quarter
Submission a paper in an International Journal	5 Quarter
Interim Report #3 documenting in brief the object-oriented project of the Attack Simulator software.	End of 6 Quarter
A-4. Development of the software prototype of the Attack Simulator implementing theoretical results of research and its evaluation.	6-9 Quarters
Demonstration of the software components that will be used in the Attack Simulator software. (On demand of US AFRL/ID.)	9 Quarter
Final Report	End of 9 Quarter

Notice: The grey shaded rows correspond to the tasks to be solved during the second year research. The Task A-4 had to be solved partially.

Objective

The main objective of Task # 1 of the project is the development of a formal model and software tool prototype for simulation of a wide spectrum of distributed attacks and also exploration of its capabilities and usefulness as applied to the computer network assurance area.

Expected results

The main expected result will be a formal framework, model, architecture and software tool prototype for simulation of distributed attacks against computer networks. In more detail, these results consist of the following:

1. Scenario-based specification of the representative set of distributed attacks (description of a set of particular cases of attacks specified on the macro-level);
2. Technique for case-based regenerating (inductive recovery) of the formal grammar specifying formal model of the attack of the given class;
3. Stochastic model of a fragment of the attack on the micro level;
4. Object-oriented project of the Attack Simulator–software tool prototype for simulation of attacks against the computer network;
5. Attack Simulator software prototype, results of exploration of the developed Attack Simulator and evaluation of its benefits in the network assurance system design and maintenance.

Technical approach

A distributed attack is planned on the macro-level as a partially ordered set of steps forming an attack scenario. Each step aims at achieving a particular sub-goal corresponding to a particular "simple" attack. While implementing a distributed attack at some steps the malefactor may succeed or fail. Each step can be implemented in many particular ways. The same steps can be implemented in various orders and can be repeatable. They can be implemented from different computers and targeted against different computer network resources. Each particular attack can be implemented with various sequences of commands. In other words, the diversity of attacks and ways of their implementations can be formidable.

To match the above mentioned peculiarities of attacks the following technical approach is used within the task of interest.

At the higher (macro-) level, the attack scenarios are formalized in terms of structured set of formal grammars interconnected with the "grammar substitution" operations. Each attack realization is specified as sequence of steps at the macro-level. Each such a sequence is interpreted as a "word" of a formal language specified formally in terms of a formal grammar. The set of "words" can be used as a training sample for inductive recovery of such a grammar. On the other hand, it is possible to use expert-based approach to formal grammars recovery and exactly this approach has yet been employed. This way was found to be the most appropriate because of lack of representative samples of the representative diversity of attacks. The analysis performed and the results obtained proved that this way of action is appropriate. The analysis of the complexity of the grammars which are capable to adequately specify the attacks against computer network also proved that one can restrict himself by usage of grammars which are no more complex than attributive (stochastic) *LL(2)*-grammars.

At the lower (micro-) level each step of an attack is detailed in terms of a sequence of events (system calls, OS commands, etc.). Event of this sequence realize a particular action of a malefactor. In fact, this level of attack model details can also be formalized in terms of formal grammars with substitution terminals by sequences of commands.

1.7. Technical progress during the first year (for 2nd annual reports)

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

Achievements of the first year

The basic achievements correspond to the tasks scheduled. These tasks and respective results are described below.

1. Development of the specification of the representative set of distributed attacks defined on the macro-level.
2. Development and selection of mathematical methods and techniques realizing the attack modeling.

3. Development of the object-oriented project of the Attack Simulator–software tool prototype for simulation of attacks against the computer network.

The main results obtained during first year research are presented below.

1. The analysis and classification of the known attacks against single computers and also against computer network on the whole was carried out. A large number of computer attacks were analyzed. The taxonomy of attacks which was used as a basis to form the representative set of attacks to be modeled and simulated was developed.

2. The conceptual description of the representative set of the basic attack classes is developed. These attacks are considered as the components of distributed attacks forming their scenarios. The scenario-based models of eight network attack classes are determined. These classes are:

(1) Analysis of the network traffic,
(2) A network scanning (probing),
(3) A substitution of the trusted object of the network and transmission of the messages from its name with appropriation of its access rights,
(4) An implantation of the false object in a network,
(5) A denial of service,
(6) An unauthorized access from a remote machine by guessing password,
(7) An unauthorized access to local super-user (root) privileges, and
(8) A remote initiation of applications.

Each scenario is described by a set of admissible sequences of steps determining an attack class on macro and micro level. Each attack class scenario is illustrated by a lot of particular examples specialized on the basis of concretization of attacked computer specific software.

The models of the "*analysis of the network traffic*" attacks include a sequence of the following stages: determination of the place in the network where from it is favorably to listen to the network; determination of analyzed levels of network protocols and the protocols; determination of the active network equipment in the network and mechanisms of its working; determination of software for analysis and OS under whose control analysis will be realized; adjustment of software and the development of rules (patterns) on whose basis information will be filtered; analysis and choice of host masking means, when an intruder analyzes the traffic; intrusion in the network and starting of all software (both analyzing the network traffic, and masking the intruder); reception and analysis (filtering) of the traffic passing through the intruder's network; disconnecting from a network; analysis, decoding, and classification of received information by the intruder.

The most important stages, which can be presented in *network scanning,* are the followings: selection of an "agent" computer and connection to it; finding computes, which are presented in the target network; recognition of the target network structure; recognition of the services started on the target computer; getting additional information about the target network.

The common stages of the attack "*substitution of the trusted object of the network*" *are*: the preparatory stage, which is concerned with an analysis of the attacked objects and substitution of information on the server; listening to the network; sending of a query (a storm of queries); sending of a reply, mathematical prediction of the next message number and its sending to the attacked host, rerouting the query on the intruder's host by the server; commands execution on the attacked host; reception and analysis of intercepted information; influencing on intercepted information; transfer of intercepted information (probably changed or substituted); distribution of attack against other objects.

The models of the *"implantation of the false object of the network"* attacks include the following *stages*: studying the attacked host network; listening to the network; sending a false message (or a storm of messages); reception and analysis of intercepted information by the intruder or the "deceived" server; influencing on intercepted information; transferring of intercepted information (probably changed or substituted).

The models of the "*denial of service*" attacks contain a sequence of the following generalized stages: reconnaissance of the network; an installation of master-agents and daemon-agents on the intermediary (auxiliary) hosts; sending messages from daemon-agents to master-agents (for example, on a status); sending information on a status of daemon-agents from master-agents to a malefactor; sending commands from the malefactor's host to master-agents; sending commands from master-agents to daemon-agents; sending a specially crafted packet from the malefactor's host (or from the host used by the malefactor) to

the intermediary host (or a set of intermediary hosts); the intermediary host (or a set of intermediary hosts) receives the packet and responds by sending a packet to the target host; the target host receives the packet and responds back to the intermediary host; sending a specially crafted packet (a sequence of the packets, fragments of the packet) from the malefactor's host (from the host used by the malefactor or by daemon-agents) to the target host.

The main stages of the "*unauthorized access from a remote machine by guessing password*" attacks are: getting information about the target system and its authentication subsystem; getting information about users of the target system; an interception of ciphered (or hashed) passwords; getting database with ciphered (hashed) passwords; single entering of the password in online mode; a multiple entering of passwords in online mode; a retrieval of the passwords in offline mode; an interception of passwords in clear text form (may be used for some other services).

The following stages are common for the "*unauthorized access to local superuser (root) privileges*" attacks: an analysis of the attack targets; a preparation of the code; an implantation of the code; an implantation of parameters (parameterization of the code); a transfer of control to the code.

The "*remote initiation of an application*" attacks are characterized by the following stages: a reconnaissance of the target computer system; an implementation of a malicious code or program text into the target system; an unauthorized access to the system resources; an initiation and usage of auxiliary software, which is legally installed in the target system; an initiation of a malicious program; an activation of some special functions, available in the implemented malicious program; sending information from the implemented program to the intruder; cleaning log files and deleting other attack evidences; a self-reproduction of a malicious program.

3. The thorough study of formal frameworks that can potentially be used as a formal mechanism for distributed attack modeling and subsequent simulation proved that formal grammar framework matches the attack modeling domain in the best way. Formal grammar mathematical framework is used for formal specification of real objects or sets of objects of regular structure. This conclusion is also applicable to the task of specification of scenarios of complicated network attacks. Therefore, the scenario of an attack against a computer system may be adequately represented in terms of formal grammar or a family of nested grammars. Besides, this grammar may play a dual role, namely it may be used both as a model of attack cases generation and as a model of attack detection based on a syntax analyzer that corresponds to a grammar used in model formal specification.

4. A thorough analysis of the methods of synthesis (recovery) of grammars applied to attack modeling task was carried out. Formally, the task of synthesizing a grammar consists in creating the algorithm to recover its syntactic structure based on the finite set of words of the language that specifies attack cases, and on the finite set of words from the supplement to this language. The grammar that generates scenarios for computer network attacks can be synthesized: (1) via inductive recovery based on the set of cases through the use of formal methods; (2) by an expert who possesses knowledge of the malicious party's intentions and the possible ways these intentions can be realized; (3) through combining the two above methods.

Two groups of algorithms are selected for grammar recovery: (1) enumeration grammar recovery algorithms; (2) induction grammar recovery algorithms. The inductive grammar recovery methods look the most adequate for the purposes of recovering grammars that specify computer network attacks; specifically, the inductive method for recovering regular grammars on the basis of positive examples (the Feldman method). This method consists in constructing a non-recursive grammar that creates precisely those strings that were present in the training example, and then arriving at a simpler recursive grammar that generates all the strings of the positive examples and an infinite number of other strings.

In order to verify the performance capabilities of the above algorithms several cases of computer network attacks have been looked up. These cases set the basic methods of implementing attacks of the following types: network scanning for identification of hosts; network scanning for identification of services; identification of operating system; shared resource enumeration; users and groups enumeration; applications and banners enumeration; actions on getting access to resources; denial of service attacks.

The examples of using grammar recovery algorithms for specification of computer network attacks were developed. These examples have proved the practical applicability of the algorithms represented here for specification computer network attacks. The synthesized grammars can generate the versions of attacks that served the training data for the development of the grammar. The grammars developed through

combining a number of productions are capable of generating the attacks that were not included in the training cases. This expands the capability of the attack simulator that is based on the utilization of these grammars.

5. A multi-layer formal model of the representative set of distributed attacks specified in terms of a family of interacting context-free attributed grammars is developed. Such a model allows to specify and to simulate distributed attacks at various levels of details. The formal model includes specifications of all its basic components. They are as follows:

- *Basic notions and components of distributed attack specification*. They include specifications of scenarios of distributed attack and malefactors' intentions. In the developed model the malefactor's intention-centric approach to the specification of its activity is used. This means that basic notions of the domain correspond to the malefactor intentions and all other notions are structured according to the structure of intentions.
- *Ontology of the "Network Attacks" domain*. It interacts with the family of the aforementioned formal grammars and each notion of the ontology corresponds to a symbol of the sequence specifying attack scenario.
- *Formal model of the attacked computer network*. In this model the attacked network is considered as an environment that reacts on the malefactors' actions. *The main attributes* taken into account in the host model are: *IP*-address, mask of the network address, type and version of OS, users' identifiers, their domain names, passwords, users' security identifiers (SID), domain parameters, active ports of the hosts (services used, busy TCP and UDP ports, etc.), running applications, security indices, shared resources, trusted hosts and some other attributes.

Under implementation angle, distributed attack is considered as a sequence of coordinated actions of spatially distributed malefactors. The developed architecture of the attack simulator implementing the above described attack model is being built as multi-agent system (MAS). Each malefactor is mapped as an intelligent agent of the same architecture and possessing similar functionality. While developing an attack, they interact via message exchange informing each other about current state and results of the attack in order to coordinate their further activity. These messages represented in standard MAS communication languages, i.e. in KQML, that is standard of DARPA (for message “wrapper”), and XML (for message content). The design and implementation of the attack simulator in being carried out on the basis of the Agent System Development Kit that was developed by the authors.

6. During the reported period the partial object-oriented design of the macro-level components of the Attack simulator was fulfilled.

The specifications of the following components are developed:

(1) Model of the malefactor’s actions;
(2) Model of the computer network under attack and the host models;
(3) Model for calculating the probability of successful actions (attacks) against the host;
(4) Model of host response to malefactor’s actions.

1.8. 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 respective results are described below.

1. Development of the object-oriented project of the Attack Simulator–software tool prototype for simulation of attacks against the computer network.
2. Development of the software prototype of the Attack Simulator implementing theoretical results of research and its evaluation.

Main results obtained during the year of reference are presented below.

1. *The approach for modeling and simulation of attacks against computer network has been defined more exactly*.

The conceptual investigation of the attack has discovered the following *peculiarities of planning and execution of attacks*, influencing upon the choice of formal model of attacks and the Attack Simulator design:

- any attack is directed to the concrete object and, as a rule, has quite a definite intention;
- an attacker's intention can be represented in terms of partially ordered set of lower-level intentions and actions realized in multiple ways; attack development strongly depends on the response of the attacked computer network, choice of attack continuation is almost always non-deterministic; the attack development scenario cannot be definitely specified beforehand, since any attack depends on many uncertainties: uncertainty in choice of the attacker's intention and attack object; uncertainty in choice of attack scenario implementing the already selected intention; uncertainty of the attacked computer network response, etc.

The peculiarities of the developed approach stipulating for design of the object-oriented project of the Attack Simulator are as follows:

- malefactor's intention-centric and target-oriented attack modeling and simulation;
- multi-level attack specification in the consecution (from upper to lower levels) of "attack task (goal) and attack object → structured malefactor's intentions → malefactors actions → attacked network response";
- ontology-based attack model structuring;
- attributed stochastic *LL*(*2*) context-free grammar for formal specification of attack scenarios and its components ("simple attacks"), that are the strings of the grammar generated left to right, top to bottom with uncertainty of the choice of substitution through the second symbol;
- using operation of formal grammar substitution for specification of multi-level structure of attacks;
- state machine-based formal grammar framework implementation;
- on-line generation of the malefactor's activity resulting from the reaction of the attacked computer network.

The more exactly defined conceptual model of attacks against computer network has permitted to elaborate the object-oriented project of the Attack Simulator.

2. For development of the object-oriented project of the Attack Simulator and its implementation, *the technology and software tool called Multi-agent System Development Kit (MAS DK) have been elaborated and used*. This tool is implemented on the basis of Visual C++ 6.0, JAVA1.3 and XML programming languages.

3. *The object-oriented project of the Attack Simulator–software tool prototype for simulation of attacks on the computer network has been developed*.

The following main components of the Attack Simulator software prototype have been determined in the object-oriented project of the Attack Simulator and designed:

(1) *The component setting the subject domain ontology* (*DomainOntology*). It serves for specifying and storing the notions, notion attributes, and values of notion attributes of the subject domain of computer network attack modeling. The subject domain ontology is filled in the design-time stage by the MAS DK ontology editor. At this stage, notions (classes) of the ontology, notion (class) attributes, as well as meta-classes that unite notions into groups (they are not used in *DomainOntology*) are entered and modified through the user's interface.

(2) *The component realizing the set of attack scenarios as a family of state machines* (*AttackModel*). Through specifying the so-called "applied" state machines, *AttackModel* determines the executable computer network attack scenarios. This component is used to specify the set of different classes of computer network attacks in the form of a family of state machines. States and transitions between states are the main elements of a state machine. The scheme of description of each state machine includes the following elements: name of a state machine, its purpose and general description; identifier of the attacks ontology node to which the state machine corresponds; state machine diagram; main parameters of the state machine; parameters of transitions; transition conditions; and scripts.

(3) *The component of interpretation of the family of state machines* (*Engine*). It realizes mechanisms of operating with the "applied" state machines specified in the component *AttackModel*, and controls their behavior by an invariant meta- state machine. The applied state machines function under the control of the meta- state machine. Meta- state machine creates examples of applied state machines, transforms state machines' scripts into the internal view, enables the communication between state machines' scripts and the component *DomainOntology*, executes

the scripts, delegates control to applied state machines, and eliminates the examples of applied state machines after they complete their function.

(4) *The kernel component defining the attack task specification and supporting the interaction with main components of the prototype* (*AgentLib*). The component *AgentLib* is a sort of "gateway" connecting the component *Engine* to the program components that realize specific applied tasks.

(5) *The user interface component for attack task specification* (*TargetObjectiv*). Attack specification includes intention, address of the attacked host (network), object of attack, and information about the object of attack (host or network) already known to the attacker.

(6) *The component calculating probability of the state machine transition to the next state* (*SMProb*). The component consists of three classes: (a) *Transition* responsible for each separate (unique) transition; (b) *Transitions* containing a probabilistic group that consists of exemplars of the class *Transition* that are responsible for each separate transition within this probabilistic group; and (c) *Prob_DB* responsible for working with the table *SrcProb*, the interaction with the classes of the probabilistic group and the classes of the separate transitions within the probabilistic group.

(7) The user interface component for visualizing the process of attack realization (*AtlogView*).

The components *AgentLib, TargetObjectiv, SMProb* and *AtlogView* are the program realization of the prototype in VC++. The components *DomainOntology* and *AttackModel* are the semantic constituents of the prototype. They are described in terms of the structures represented by the MAS DK environment. The object-oriented description of the software realization of the state machine model and the interaction between the state machine model and the subject domain is contained in the component *Engine*.

4. Based on the decisions elaborated in the object-oriented project *the first version of the Attack Simulator prototype has been developed*.

The created object-oriented project has indicated that *further development of the* Attack Simulator software prototype will imply some serious changes.

All *revisions and additions to the prototype* can be divided into two groups, based on the terms of their realization:

(1) immediate (short-term) and

(2) strategic (long-term) (in most solutions, these changes go beyond the constraints of this project).

The following are the *immediate* solutions:

- Inclusion into the prototype of the possibility to initiate the attack modeling process when part of the data regarding the attacked system is known.
- Inclusion into the prototype the possibility to specify concrete objects of attack (files, resources, launched applications, accounts, etc.).
- Expanding classes of attacks.
- Realizing certain classes of attacker's actions on micro-level.
- Realization of taking into account the structure of the attacked network and the specific features of hosts in modeling the attacks.
- Specifying the structure of the attacked network (networks) through a user interface.
- Introducing additional specification of hosts in the attacked networks for recognizing a host type.

The following are the main strategic solutions in the development of the prototype:

- Using a real network as an attack object.
- Using real utilities and exploits in realizing attack scenarios.
- Realization of the attack modeling system as a team of hacker-agents that are collectively solving a common task.

5. *The second version of the Attack Simulator prototype has been developed*.

The following subtasks were realized in this version of the Attack Simulator prototype:

- Specifying the structure of the attacked network (networks) through a user interface.
- Realizing attacker's actions on micro-level.
- Introducing additional specification of hosts in the attacked networks for recognizing a host type.
- Inclusion into the prototype the possibility to initiate the attack modeling process when part of the data regarding the attacked system is known.

The implementation of the two first subtasks has appeared most labour-consuming.

The user interface to specify the network structure was realized as a wizard. At the first stage of the attacked network model specification, all subnets contained in the structure of the attacked network are specified (by using the notion *LAN* of the *DomainOntology*). For each subnet, its IP-address, name, and mask are specified. At the second stage of the attacked network model specification, connections between subnets and the location of the attacker in relation to the attacked network are determined. The connections are specified through a connectivity matrix. At the third stage, the hosts included in each subnet are specified, as well as its relative location on the connectivity matrix for each subnet in the order of their specification by the user at the first stage. After the specification of all hosts and all subnets, the information is mapped into the corresponding tables of the attacked network's input database.

Solution of the subtask of *realizing attacker's actions on micro-level* was achieved by creating several classes of message templates unified by their association with data transfer protocols and operating system command types. Templates correspond to the headers of packets in the format of the protocols used in the realization of the attacker's specific action (terminal from the standpoint of state machine model). The templates are assigned the necessary parameters in the corresponding terminal states of the state machine model. For example, to form a message based on the header of a TCP packet header template, the following parameters are specified: time, sender address, destination address, name of protocol, value of the flag, etc. In point of fact, realizing micro-level actions implies expanding the terminal states in which attacker interacts with attack objects (host, network, networks) to the level that approximates (to a specified degree) real attacks on computer networks – level of messages within the used data transfer protocol.

On the side of the attacked network (host) model, the messages are processed with regard to the following information:

- Attack object features – if it is a host, whether it is a firewall or a server; if it is a network (a set of networks), whether there are hosts on the attack route that may influence upon the further development of the attack (here, mostly firewalls are implied).
- Attack object parameters – presence of various open ports, running applications, availability of certain resources, etc.

The response message from the attacked network (host) model is formed with respect to the following:

- Accessibility of the attack object, based on the network structure.
- Logical-probabilistic model of the attacker's specific action.
- Attack object's degree of protection (considering the host model).

6. *The third version of the Attack Simulator prototype has been developed*. This version of the Attack Simulator software prototype is realized as two cooperating agents – agent *MainHack* (hacker-agent) and agent *MainNet* (attacked computer network agent).

The agent *MainHack* is based on the state-machine probabilistic model of transitions between attack actions. This model is an interpretation of the formal grammars-based attack generation approach, offered by the Project authors. Information base of the agent *MainHack* is the repository of the information received from the agent *MainNet* .

The agent *MainNet* is based on the probabilistic-attributive scheme of reaction to the hacker-agent *MainHack* actions. The agent *MainNet* outputs the network (host) information in reply to the message-request (attack action) of the agent *MainHack* with some probability depending on the network (host) configuration, running applications, versions and types of operating systems, and also network security settings. The success probability at the agent *MainNet* part is calculated only in case of satisfaction of all conditions for the hacker's attack action, directed to an attacked host (if the attack goal is a host) or to a set of hosts (if the attack is directed on a network).

The communication between the agents *MainHack* and *MainNet* in the software prototype is based on the use of the concept "Attack" of the ontology "Attacks on computer networks". In the current version of the Attack Simulator software prototype there are 27 attack attributes, including the attributes which determine a result of attacks against DNS-server.

The incoming messages are processed by appropriate scripts. The processing is carried out both at the hacker-agent *MainHack* part, and at the agent *MainNet* part. Thus the agent *MainHack* performs the executed action registration and the subsequent action definition. After reception of the input message (a

packet of the input messages) the agent *MainNet* performs their processing and generation of the response directed to the agent *MainHack*.

For separate classes of attacks (therefore and for messages) both at the hacker-agent *MainHack* part and the agent *MainNet* part the specialized state-machines (with names "a class/subclass of attack + _ MSG") were realized. These state-machines are responsible only for the communications between the agents.

After transformation to the two-agent architecture of the prototype, the processing of conditions of attack actions execution was excluded from the agent *MainHack* scripts. It was realized in a separate program component of the agent *MainNet*. Thus, in the present version of the prototype at an attack realization the hacker-agent has only two types of information on the attacked network: (1) specifications of the attack goal (task); (2) results of the previous attacks.

1.9. Current technical status

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

1.10. Cooperation with foreign partners

According to the Work plan, one Interim Report (Interim Report #3) was submitted to the Partner (by June 1, 2002). It contained the results of all predefined research. The Project executors together with the Partner representatives participated in the workshop, organized by the Partner to discuss the research results of the first year in March 2002 at Binghamton University, USA.

1.11. Problems encountered and suggestions to remedy

None

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

These perspectives will be discussed at the meeting that is planned to be held in April 2003. Proposal for continuation of the research supposed by this Project is submitted to the Partner in July 2002.

Attachment 1: Illustrations attached to the main text

None

Attachment 2: Other Information, supplements to the main text

Brief content of the Interim reports submitted to the Partner

Interim Report #3

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

1. Gorodetski V., Karsayev O., Kotenko I., Khabalov A. Software Development Kit for Multi-agent Systems Design and Implementation // Lecture Notes in Artificial Intelligence, Vol.2296, Springer Verlag, 2002. P.121-130.

   Abstract. The paper presents the developed technology and software tool for design and implementation of knowledge-based multi-agent systems. The software tool comprises two components that are "*Generic Agent*" and "*Multi-agent System Development Kit*" (MAS DK). The former comprises reusable Visual C++ and Java classes and generic data and knowledge base structures, whereas the latter comprises several developer-friendly editors aimed at formal specification of the applied multi-agent system (MAS) under development and installation of the resulting application in particular computer network environment. The developed technology and MAS DK were used in the design and implementation of the MAS prototype for computer network assurance and intrusion detection and distributed attack simulator.

2. Gorodetski V., Kotenko I. Attacks against Computer Network: Formal Grammar-based Framework and Simulation Tool // A.Wespi, G.Vigna, L.Deri (Eds.). Recent Advances in Intrusion Detection. Fifth International Symposium. RAID 2002. Zurich, Switzerland. October 2002. Proceedings. Lecture Notes in Computer Science, V.2516. P.219-238.

   Abstract. The paper presents an approach and formal framework for modeling attacks against computer network and its software implementation on the basis of a multi-agent architecture. The model of an attack is considered as a complex process of contest of adversary entities those are malefactor or team of malefactors, on the one hand, and network security system implementing a security policy, on the other hand. The paper focuses on the conceptual justification of the chosen approach, specification of the basic components composing attack model, formal frameworks for specification of the above components and their interaction in simulation procedure. The peculiarities of the developed approach are the followings: (1) malefactor's intention-centric attack modeling; (2) multi-level attack specification; (3) ontology-based distributed attack model structuring; (4) attributed stochastic *LL(2)* context-free grammar for formal specification of attack scenarios and its components ("simple attacks"); (5) using operation of formal grammar substitution for specification of multi-level structure of attacks; (6) state machine-based formal grammar framework implementation; (7) on-line generation of the malefactor's activity resulting from the reaction of the attacked network security system.

3. Gorodetski V., Kotenko I. The Multi-agent Systems for Computer Network Security Assurance: frameworks and case studies // IEEE ICAIS-02. IEEE International Conference "Artificial Intelligence Systems". Proceedings. IEEE Computer Society. 2002. P.297-302.

   Abstract. The paper presents experience in application of multi-agent technology for design and implementation of multi-agent systems (MASs) intended to cooperatively solve the currently critical tasks in the area of computer network security assurance. These MASs 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 technology and software tool "Multi-agent System Development Kit" developed by authors. The paper sketches the above MASs and analyses advantages of use of multi-agent architecture for computer network assurance.

4. V.Gorodetski, I.Kotenko. Formal Model of complex distributed attacks on Computer Networks. II Inter-regional Conference "Information Security of Russia Regions". Proceedings. Vol.2. Saint-Petersburg, 2002. P.92-97. (in Russian)

Abstract. The paper considers a formal model of complex distributed attacks on computer networks. The developed formal model is constructed as hierarchy of context-free attribute stochastic grammars connected by substitution operation. The offered formal model can be used as a basis of the system intended for an experimental estimation of computer network protection. It can play an important role at analysis of the security policy efficiency.

5. V.I.Gorodetski, I.V.Kotenko. Teamwork of Agents in Antagonistic Environment. International Conference on Soft Computing and Measurements. SMC'2002. Proceedings. Saint-Petersburg, 2002. Vol.1. P.259-262. (in Russian)

Abstract. The agents form the agents' team, if they apply joint efforts for reaching a common goal and function in the dynamic external environment under conditions of the adversary's counteraction. In the paper the issues of the agents' team construction and teamwork in the antagonistic environment are considered. The high-level models of teamwork are determined. As an example of the offered approach implementation the case study intended for simulation of joint work of the hackers-agents is described.

6. Kotenko I., Stankevich L.A. The Control of Teams of Autonomous Objects in the Time-Constrained Environments // IEEE ICAIS-02. IEEE International Conference "Artificial Intelligence Systems". Proceedings. IEEE Computer Society. 2002. P.121-130.

Abstract. The paper considers problems of control of teams of autonomous objects in the time-constrained or real-time environments. The generic models of teamwork are determined. The common approach to support of teamwork in conditions of temporary limitations based on combination of the approximation calculations and anytime-algorithms is offered. The models of the agents' teamwork realizing this approach for three different application areas are described: (1) virtual soccer (Robocup), (2) simulations of battle operations realized by a group of autonomous flight vehicles, (3) simulations of distributed coordinated computer attacks fulfilled by the hackers' team.

7. I.Kotenko, E,Man'kov. Simulation of Attacks on Telecommunication Systems. VIII International Conference on Informational Networks, Systems and Technologies. ICINSAT-2002. SUT. St.Petersburg, 2002. P.190-198. (in Russian)

Abstract. The paper considers an approach to construction and program implementation of the system for simulation of the program attacks against telecommunication systems. The attacks simulation system is intended for testing of telecommunication systems safety, lowering of time and cost expenditures on analysis of the protection mechanisms efficiency. The attacks simulation system is constructed on the basis of the ontology "Modeling and simulation of attacks on telecommunication systems", a formal model of the distributed attacks as stochastic attribute context-free grammars, its computing interpretation as a set of nested state machines, and usage of the MAS DK toolkit for multiagent systems development.

8. I.V.Kotenko. Multi-agent Technologies for Support of Intrusion Detection in Computer Networks. X Russian Conference "Methods and tools of information assurance". Proceedings. Saint-Petersburg, SPbSPU. 2002. P.44-45. (in Russian)

Abstract. The paper presents the basic applications of multi-agent systems in the field of intrusion detection in computer networks developed in Intelligent Systems Laboratory of SPIIRAS: (1) agent-based simulator of attacks on computer networks; (2) multi-agent intrusion (attack) detection system; (3) multi-agent system for intrusion detection learning in computer networks. Each of the developed applications is founded on the formal models and architectures offered by the scientists of the laboratory, and is realized by use of own toolkit for multi-agent system development MAS DK ("Multi-Agent System Development Kit").

9. V.I.Gorodetski, I.V.Kotenko. Teamwork of Hackers-Agents: Application of Multiagent Technology for Simulation of Distributed Attacks on Computer Networks. VIII National conference with international involvement on Artificial Intelligence. Proceedings. Moscow, 2002. P.711-720. (in Russian)

Abstract. The paper considers an approach to the agents' teamwork implementation. This approach is described on an example of simulation of the coordinated distributed attacks on computer

networks fulfilled by a group of hackers-agents. The approach is based on main positions of the "joint intentions" theory and the "common plans" theory. The offered technology of creation of the agents' team includes the following stages: (1) formation of the subject domain ontology; (2) determination of the agents' team structure and mechanisms of their interaction and coordination; (3) specifications of the agents' actions plans as a hierarchy of attribute stochastic formal grammars; (4) assignment of roles and allocation of plans between the agents; (5) state-machine based interpretation of the teamwork. The stages of ontology creation, agents' plans specification and state-machine based interpretation of attack generation are considered. The prototype of multiagent system for attack simulation is described.

10. I.V.Kotenko. Taxonomies of attacks on computer systems. SPIIRAS Proceeding, Vol.1. SPb, SPIIRAS, 2002. (in Russian)

    Abstract. In the paper, a review of the taxonomies of the attacks on computer systems is presented. The analysis of the following types of the taxonomies are fulfilled: lists of attack terms; lists of attack categories; attack results categories; empirical lists of attack types; vulnerabilities matrices; action-based taxonomies; taxonomies based on the attack signatures; security flaws or vulnerabilities taxonomies; incident taxonomies.

11. I.V.Kotenko. Case-based Recovering of Formal Grammars specifying Scenarios of Computer Attacks. Third International Conference "Artificial Intelligence -2002". Proceedings. 2002. Crimea, Ukraine, 2002. (in Russian)

    Abstract. The paper analyzes the following approaches for synthesis of the formal grammars, specifying the scripts of computer network attacks: (1) inductive recovery on a set of precedents by means of formal methods; (2) definition by the expert on the basis of knowledge of malefactor's intentions and possible ways of its realization; (3) combination of the first and second ways. The examples of the attack grammar recovery algorithms are submitted.

12. I.V.Kotenko. Case-based Recovering of Formal Grammars specifying Scenarios of Computer Attacks. International magazine "Artificial Intelligence", № 3, 2002. (in Russian)

    Abstract. In the paper, the approach for case-based synthesis (generation) of formal grammars specifying models of attacks against computer networks is considered. Two groups of grammar recovery algorithms are selected: enumeration grammar recovery algorithms and induction grammar recovery algorithms. The inductive method for recovering regular grammars by a positive example (the Feldman method) has been chosen as the most appropriate. The description of its algorithmic implementation has been included. In order to demonstrate the performance capabilities of grammar recovery algorithms used to describe attacks on computer networks, several cases of computer network attacks are reviewed. The example of using grammar recovery algorithm for specification of computer network attacks is elaborated. This approach is intended for realization of multi-agent system of computer network attacks simulation.

13. Kotenko I.V., Alekseev A.S. Simulation of Distributed Denial of Service attacks based on teamwork of software agents. Regional conference "Regional informatics-2002". Proceedings. SPb., 2002. P.93-94. (in Russian)

    Abstract. The paper considers the approach for organization of software hackers-agents' joint work based on mechanisms of multi-agents' teamwork. This approach is presented on an example of the task of the DDoS (Distributed Denial of Service) attacks simulation, such as Trinoo, TFN, Stacheldraht, etc. The common technology of the agents' teamwork for imitation of DDoS-attacks is offered. This technology includes the following stages: formation of the DDoS-attacks ontology; definition of the agents' team structure; development of mechanisms for agents' interaction and coordination; specification of hierarchy of the DDoS-attacks realization plans; assignment of roles and distribution of the plans between the agents.

14. Kotenko I.V., Stepashkin M.V. Classification of attacks on Web-server. Regional conference "Regional informatics-2002". Proceedings. SPb., 2002. P.134. (in Russian)

    Abstract. The paper offers a classification of attacks on Web-server. This classification is based on stages of attack realization. It is used for description of the "Attacks on Web-server" ontology. The attacks on Web-server are allocated into the groups ensuring (1) investigation (reconnaissance), (2) implantation, (3) increases of privileges, (4) threat realization; (5) covering tracks; (6) creating back doors.

15. Kotenko I.V., Nesterov S.A. Approach to building of network attack source model using formal grammar apparatus. Regional conference “Regional informatics-2002”. Proceedings. SPb., 2002. P.124-125. (in Russian)

Abstract. The paper describes the task of detection unknown network attacks on computer systems. The model of the network attack source is considered. It is founded on the basis of formal grammars. The ways of use of this model for intrusion detection are determined. The approach for application of the developed model for the risk analysis in sphere of computer network security is defined.

IV. SIGNATURES

Task #1 Principal Investigator
Professor Igor V.Kotenko,
Doctor of Sciences (Tech), Ph. D.