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

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

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

5. Commencement Date, Duration

December 1, 2000, 27 months

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 third year research.

Objective

The main objective of the Task 1 of the project is 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 is a formal framework, model, architecture and software tool prototype for simulation of distributed attacks on computer networks. In more detail, these results are as follows:

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 on 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 at macro-level as a partially ordered set of steps forming an attack scenario. Each step aims at achieving of 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 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 found out 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 grammars which are no 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.

7. Technical progress during the second year

Technical progress during the second year was 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 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 on the computer network.
2. Development of the software prototype of the Attack Simulator implementing theoretical results of research and its evaluation.

The main results obtained during the second year 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 on choice of a formal model of attacks and the Attack Simulator design: any attack is directed on the concrete object and, as a rule, has a quite 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 greatly 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) "attack task (goal) and attack object $\rightarrow$ structured malefactor's intentions $\rightarrow$ malefactors actions $\rightarrow$ 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 is the strings of the grammar are 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:

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 interface.

The component realizing of 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 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.

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 between applied state machines, and eliminates the examples of applied state machines after they complete their function.

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 a "gateway" connecting the component Engine to the program components that realize specific applied tasks.

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.

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 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.

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 Attack Simulator prototype has been developed*.

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 on 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 on the agent *MainNet* part is calculated only in case of satisfaction of all conditions for the hacker's attack action, directed on an attacked host (if the attack goal is a host) or on 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 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 on DNS-server.

The incoming messages are processed by use of appropriate scripts. The processing is carried out both on the hacker-agent *MainHack* part, and on 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 a their processing and generation of the response directed to the agent *MainHack*.

For separate classes of attacks (therefore and for messages) both on the hacker-agent *MainHack* part and 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 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 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 the information on the attacked network: (1) specifications of the attack goal (task); (2) results of the previous attacks.

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 software prototype of the Attack Simulator implementing theoretical results of research and its evaluation.
2. Simulation-based exploration of the developed Attack Simulator and its benefits in the network assurance system design and maintenance.

The *main results* obtained during the year of reference are presented below.

1. *The developed Attack Simulator prototype has been modified*.

The software prototype for computer network attack simulation is built as a multi-agent system that uses two classes of agents. The agent of the first class simulates attacked computer network and defense system of the network ("Network Agent"). The second one simulates a hacker performing attack against computer network ("Hacker Agent"). In the developed prototype each agent class has single instance although the developed technology makes it possible to simulate a team of hackers and a team of agents responsible for computer network security.

The aforementioned agents are implemented on the basis of the technology supported by Multi-Agent System Development Kit (MASDK) that is a software tool aiming at support of the design and implementation of multi-agent systems of a broad range. The developed and implemented Attack Simulator comprises the multitude of reusable components generated by use of the MASDK standard functionalities and application-oriented software components developed in terms of programming language MS Visual C++ 6.0 SP 5.

Each agent operates using the respective fragment of the application ontology that is designed by use of an editor of MASDK facilities. The interaction between agents in the process of attack simulation is supported by the communication environment, which design and implementation is also supported by MASDK.

It is noteworthy to note that the first version of the prototype was implemented as a system that consisted of a single agent simulating a hacker's activity whereas computer network security system was simulated as a reactive system. In such architecture there were no need in using communication environment by both attacking Hacker Agent and computer network security reactive system. In the current version of the prototype, the communication component plays a very important role. Indeed, the knowledge bases of Network Agent and Hacker Agent are implemented as two separate entities. Such a knowledge representation makes it actually possible to simulate adversary interactions. In it, while simulating an attack in order either to obtain response providing it with the needed information from the Network Agent (on the reconnaissance stage) or to perform an attack action (on the threat realization stage) Hacker Agent sends a certain message to the Network Agent. The Network Agent, like this takes place in real-life interactions, analyzes the received message and forms a responsive message. This message is formed based on the Network Agent's knowledge base that models the network configuration and all its attributes needed to simulate the real-life response. The Network Agent's knowledge base also uses information about possible existing attacks and reaction of the network on them.

The key components of both agents correspond to so-called kernels that are the modules written in C++ and compiled into a dll. These components provide interface between the part of the software written in C++ and the components implemented through use of MASDK. The kernels provide interfaces to the respective fragments of the application ontology, and initialize the state machines, which in turn execute their scripts.

The Hacker Agent comprises the following main components: (1) Agent hacker Kernel; (2) fragment of the application domain ontology; (3) state machines model; (4) scripts; (5) attack task specification component; (6) probabilistic (stochastic) decision making model with regard to the further actions; (7) network traffic generator; (8) visualization component of the attack scenario development.

The kernel of the Hacker Agent (Hacker.dll) contains a standard set of functions needed for exploiting ontology and running state machines. It is also provided with functions that call specification of attack task, compute next state-machine transition as well as initiate and perform visualization of the attack development.

Fragment of the application domain ontology specifies a set of notions and attributes used by the Hacker Agent.

State machines model component is used for specification of the Hacker Agent behavior including decision making mechanism used by the Hacker Agent's to choice the next steps of action. The state machine model of the Hacker Agent is built on the basis of attribute stochastic grammars, and consists of over 50 nested state machines.

Script component specifies the set of scripts that can be performed by the Hacker Agent's state machines.

Attack task specification component provides user with interface needed for him to specify attack attributes.

Probabilistic decision-making model is used to determine the Hacker Agent's further actions in attack generation.

Network traffic generator is used to form the flow of network packets for several classes of attacks directed to the hosts according to the attack specification. This component is initiated through calling the particular kernel function of the scripts of those states, for which the network traffic has to be generated.

Visualization component of the attack scenario development is used for visual representation of the attack progress, corresponding to each action of attacker and respective response of the Network Agent. The response may be effective (the attack action was successful in part or in full), or ineffective (no response message, or the message saying that the attack was blocked by the firewall).

The *main components of the Network Agent* are as follows: (1) Network Agent Kernel; (2) fragment of the application domain ontology; (3) state machines model component; (4) scripts component; (5) network configuration specification component; (6) firewall model (implementation) component; (7) generator of the network’s response to attack action.

Network Agent Kernel (NetAgent.dll) contains the standard set of functions for processing the application domain ontology and the state machine model, as well as the functions used for specification of network configuration through user interface, for the firewall model initialization, and for computation of the network’s response to an attacking action.

Fragment of the application domain ontology determines a set of notions and attributes used by the Network Agent.

State machines model of the Network Agent's specifies its behavior. This state machines model consist of single state machine that mostly performs communication functionality. This component specifies the actions corresponding to the incoming message receiving, their classification, processing, and sending the response.

Scripts component specifies a set of scripts initialized from the state machine model of the Network Agent.

Network configuration specification component is used for the specification of a set of user interfaces for the description and configuration of the network to be attacked. All the notions and attributes that pertain to network and hosts, including the notions and attributes that describe firewalls, are attributed through this interface.

Firewall model (implementation) component is used to determine the firewall’s response to the action generated by the Hacker Agent. Each incoming message from the Hacker Agent that constitutes an attack action is entered into the firewall model, which is assigned to the entire network (in imitation of a network firewall) or (and) the attacked host (in imitation of a personal firewall). In the event of an attack being blocked by the firewall, the response message formed by the Network Agent contains only the information that the attack has been blocked by a specific firewall.

Generator of the network’s response to attack action is used for the generation of the network’s and hosts’ responses (messages) to attack actions. It is initialized through the corresponding function exported by the agent’s kernel after the Hacker Agent has successfully overcome a firewall.

2. *The simulation-based exploration of the developed Attack Simulator has been fulfilled.*

The main objective of the *experiments* with the Attack Simulator prototype has consisted in demonstration of its efficiency for various specifications of attacks and an attacked network configuration. The authors of the Project had the purpose to investigate the Attack Simulator prototype opportunities for realization of the following tasks:

(1) *Checking a computer network security policy at stages of conceptual and logic design of network security system*. This task can be solved by simulation of attacks at a macro-level and researches of responding a being designed (analyzed) network model;

(2) *Checking security policy (including vulnerabilities recognition) of a real-life computer network*. This task can be solved by means of simulation of attacks at a micro-level, i.e. by generating a network traffic corresponding to real activity of malefactors on realization of various security threats.

Therefore all fulfilled experiments have been divided into two classes:

(1) *Experiments on simulation of attacks on macro-level*. In these experiments, generation and investigation of malicious actions against computer network model were carried out;

(2) *Experiments on simulation of attacks on micro-level*. In these experiments, generation malicious network traffic against a real computer network was fulfilled.

In the experiments with simulation of attacks on macro-level, explorations of attacks for all malefactor's intentions implemented have been accomplished. These experiments were carried out for various parameters of the attack task specification and an attacked computer network configuration. Besides malefactor's intention, it was investigated the influence on attacks efficacy of the following *input parameters*:

- protection degree of network and personal firewall,
- protection degree of attacked host (for example, how strong is the password, does the host has sharing files, printers and other resources, does the host use trusted hosts, etc.), and
- degree of hacker's knowledge about a network.

To investigate the Attack Simulator capabilities, the following *parameters of attack realization outcome* have been selected:

- number of terminal level attack actions,
- percentage of the hacker's intentions realized successfully,
- percentage of "successful" network responses on attack actions,
- percentage of attack actions blockage by firewall,
- percentage of "ineffective" results of attack actions (when attack is not successful).

In all experiments the Attack Simulator allows to generate the clearly interpretable results.

Taking into account the Final Report volume constraints, we described the results of experiments on macro-level only for two classes of intentions concerning to each of the high-level intentions *Reconnaissance* (*R*) and *Implantation and threat realization* (*I*).

For high-level intention *R*, we presented the results of experiments for intentions *Identification of the host Services* (*IS*) and *Applications and Banners Enumeration* (*ABE*), and for high-level intention *I*, the results of experiments for intentions *Gaining Access to Resources* (*GAR*) and *Confidentiality Violation Realization* (*CVR*).

At carrying out the attacks realizing intentions *IS* and *ABE*, it was supposed, that network firewall can protect the attacked network by "Strong", "Medium" and "None" degrees of defense depending on completeness of terminal level attacks list that can be recognized by firewall. For intentions *IS* and *ABE*, the plots of the dependencies of the attack outcome parameters from the network firewall protection degree have been built.

At fulfilling the attacks realizing intentions *GAR* and *CVR*, attacks were carried out under the following varying conditions:

(1) for two values of protection degree of the network firewall (1 – "Strong"; 2 – "None");

(2) for two values of protection degree of personal firewall (1 – "Strong"; 2 – "None");

(3) for two values of protection degree of parameters of attacked host (1 – "Strong"; 2 – " Weak"); and

(4) for two values of the level of hacker's knowledge about a network (1 – "Good"; 2 – "Nothing").

For intentions *GAR* and *CVR*, the plots of dependencies of attack outcome parameters from various input parameters have been constructed.

In the current version of the prototype, the network traffic generation is only implemented for certain network attacks. Those attacks are selected from different classes of attacks and (or) malefactors' intentions specified in the application domain ontology. The authors have not tasked themselves with implementing all attack actions on lower level. The main emphasis has been made on developing the general approach to generating the network traffic by use of the attack simulator prototype and assessing its feasibility and effectiveness.

For evaluation of the efficacy of the Attack Simulator prototype at a micro-level, the network packets for the attacks classes "*Port scanning*", "*Denial of service*", and "*Password Guessing*" have been generated. The network model used in experiments with the Attack Simulator corresponded to a real computer network against which attacks at a micro-level were carried out.

The *simulation-based exploration of the developed Attack Simulator prototype* has demonstrated its efficacy for accomplishing various attack scenarios against networks with different structures and security policies implemented.

Besides, the *Final Report* has been developed.

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 *Final Report* was submitted to the Partner (to March 1, 2003). It contains the results of all predefined research.

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

Chapter 1 summarizes the suggested approach for modeling attacks against computer network, the developed technology and software tool for design and implementation of knowledge-based multi-agent systems, the object-oriented project of the Attack Simulator, and also contains a review of relevant works.

In Chapter 2, the results of research on the task A-4 are presented. The Chapter describes the architecture and main components of the Attack Simulator prototype, as well as its functional capabilities and specific features of implementation. It also outlines the results of simulation-based exploration of the developed Attack Simulator prototype.

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 is 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 Final report submitted to the Partner

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

1. 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, pp.191-200.

   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.

2. I. Kotenko, E. Man'kov. Agent-Based Modeling and Simulation of Computer Network Attacks. Proceedings of Fourth International Workshop "Agent-Based Simulation 4 (ABS 4)". Jean-Pierre Muller, Martina-M.Seidel (Editors). April 28-3,. Montpellier, France, 2003, pp.121-126.

   Abstract. The paper describes an agent-based approach and the software system for the simulation of remote attacks against computer networks (Attack Simulator) intended for active vulnerability assessment of computer network security. The suggested approach is realized via the automatic imitation of the distributed hackers' attacks of different complexity. The model of attacks is considered as a complex process of contest of adversary entities (team of malefactors) and network security system. The developed agent-based Attack Simulator is based on malefactor's intention modeling, ontology-based attack structuring, attributed stochastic grammar for specification of attack scenarios, its interpretation based on state machines and on-line generation of the malefactor's activity resulting from the reaction of the attacked security system.

3. I. Kotenko. Teamwork of Hackers-Agents: Modeling and Simulation of Coordinated Distributed Attacks on Computer Networks. Proceedings of The 3rd International/Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2003). Prague, Czech Republic. June 16 – 18, 2003. "Multi-Agent Systems and Applications III". V.Marik, J.Muller, M.Pechoucek (Editors), Lecture Notes in Artificial Intelligence, Springer-Verlag, vol.2691, pp.464-474.

   Abstract. The paper considers an approach to the agents' teamwork implementation. It 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 Attack Simulator software prototype and its evaluation results are described.

4. V.I.Gorodetsky, I.V.Kotenko, J.B.Michael. Multi-agent Modeling and Simulation of Distributed Denial of Service Attacks on Computer Networks. Proceedings of Third

International Conference “NAVY AND SHIPBUILDING NOWADAYS” (NSN’2003). St. Petersburg, Russia, June 26 – 28, 2003, pp.38-47.

Abstract. The move toward practical use of modern naval network-centric warfare (NCW) brings with it the benefits caused by applying distributed computing to gain superiority over its adversary. However, the adversary will attempt to attack information infrastructures used in NCW. One effective means of destruction of such infrastructures is the use of distributed denial-of-service (DDoS) attacks. The primary goal of such attacks is to break or reduce the availability of distributed computing resources to legitimate users, with second-order effects such as causing nodes in the infrastructure to crash or even become compromised. Increase of survivability of information systems and structures requires the development of both strict theoretical and practical basis. The availability of DDoS attacks modeling and simulation means would be a significant component of such a basis. The paper introduces a framework for modeling and software tool for simulation of a broad spectrum of DDoS attacks, which key building blocks are ontology of DDoS-attacks, mechanisms for teamwork of software agents representing the hackers performing DDoS attacks and multi-agent platform called Multi-Agent System Development Kit supporting the agent-based DDoS attack modeling and simulation technology.

5. I.Kotenko, E.Man’kov. Experiments with simulation of attacks against computer networks. Lecture Notes in Computer Science, Springer-Verlag, vol.2776. Theory and Practice of Computer Network Security. Proceedings of the International Workshop on Mathematical Methods, Models and Architectures for Computer Network Security, St. Petersburg, Russia, September 21–23, 2003, pp.187-198.

Abstract. The paper describes implementation issues of and experiments with the software tool “Attack Simulator” intended for active assessment of computer networks vulnerability at the stages of design and deployment. The suggested approach is based on malefactor's intention modeling, ontology-based attack structuring and state machines specification of attack scenarios. The paper characterizes a generalized agent-based architecture of Attack Simulator. The generation of attacks against computer network model and real computer network is analyzed. The experiments demonstrating efficiency of Attack Simulator in generating various attacks scenarios against computer networks with different configurations and security policies are considered.

6. I.Kotenko, A.Alexeev, E.Man’kov. Formal Framework for Modeling and Simulation of DDoS Attacks Based on Teamwork of Hackers-Agent. Proceedings of 2003 IEEE/WIC International Conference on Intelligent Agent Technology, Halifax, Canada, October 13-16, 2003, IEEE Computer Society. 2003, pp.507-510.

Abstract. The modern Internet is at rather dangerous stage of its life cycle. Taking into account a today's level of computer network security, the Internet can simply cease to work, if the current tendency of growth of number and capacity of distributed denial-of-service (DDoS) attacks to root servers will proceed. In the paper we discuss that in order to combat DDoS, the computer community needs to develop a strong theoretical basis upon which to harden information systems and infrastructures so they can survive such attacks. We introduce an agent-based formal framework for modeling and simulation of DDoS attacks. The framework and software tool developed can be used for conducting experiments to analyze computer network vulnerabilities and evaluate efficiency and effectiveness of security policy.

7. I.Kotenko. Active Assessment of Computer Networks Vulnerability by Simulation of Complex Remote Attacks. Proceedings of 2003 International Conference on Computer Networks and Mobile Computing (ICCNMC-03). Shanghai, China, October 20-23, 2003. IEEE Computer Society, 2003, pp.40-47.

Abstract. The paper describes the formal approach and software tool “Attack Simulator” intended for active vulnerability assessment of computer network security policy at the stages of design and deployment of network security systems. The suggested approach is based on stochastic grammar-based models of attacks and is realized via automatic imitation of remote computer network attacks of different complexity. The paper characterizes the Attack Simulator architecture and the processes of generating malicious actions against computer network model and real-life computer networks. The results of experiments that demonstrate the Attack Simulator efficiency are described in detail.