Comput Syst Sci & Eng (2003) 4: 191–11
© 2003 CRL Publishing Ltd

International Journal of
Computer Systems
Science & Engineering

Multi-agent technologies for computer network security: Attack simulation, intrusion detection and intrusion detection learning

Vladimir Gorodetski, Igor Kotenko and Oleg Karsaev

St. Petersburg Institute for Informatics and Automation, 39, 14th Liniya, St. Petersburg, 199178, Russia
Email: {gor, ivkote, ok}@mail.iias.spb.su

This paper presents experience in application of multi-agent technology to 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 agent technology supported by software tool "Multi-agent System Development Kit" developed with participation of the authors. The paper sketches the above MASs and analyses advantages of applying multi-agent technologies for computer network security.

Keywords: multi-agent technology, intrusion detection, attack simulation, network security

1. INTRODUCTION

During the last few years the computer network security remains a problem of great concern within information technology research area. Increase of network scale, development of advanced information technologies, and other factors enhance the number of possible targets for attacks against computer networks. These factors negatively influence upon the efficiency of the existing computer networks security systems and enable research and development of *new protection models and technologies*.

Along with the conventionally used security tools like firewalls, intrusion detection systems (IDSs) are becoming of supreme significance. It is well known that modern real-time IDSs are not able to detect sophisticated attacks performed by professionals. Newly invented attacks are realized as coordinated distributed operations of groups of professionals; they are carried out from different locations, through different entry points, and at different time moments. On the other hand, IDS often interprets normal operation of a computer network as hostile actions thus producing many false alarms.

A considerable improvement of IDS efficiency could be achieved in case of using knowledge obtained from generalization and formalization of accumulated experience regarding computer system vulnerabilities and attack cases. This is a cogent argument for the necessity of deep study and research of essence and peculiarities of coordinated distributed attacks. The study cannot be only restricted by generalization of the experience; it has also to be based on using of formal models of attacks and *attack simulation tools*. These models and tools could be very valuable in the design of IDS capable to operate with high-level notions like “identification of an attack scenario”, “forecasting of an attack development”, etc. Such capabilities could make feasible to break on-line an attack development before the irreversible consequences occur. Besides, attack simulation tools could play an important role in validation of security policies. Such tools could be used as testing equipment of IDS thus providing decrease of cost and time of the security policy validation.

The current state-of-the art in the IDSs area forces researchers and developers to focus on the elaboration of such systems that “would be capable to learn detection of new attacks and counter-measures in a semi-automatic mode in order to eliminate, as much as possible, the manual and ad-hoc elements from the process of building an intrusion detection system” [1]. Although *intrusion detection learning* is the problem that is researched about last ten years, nevertheless its importance is permanently growing. In order to detect attacks against a particular host or against the computer network as a whole, it is necessary to solve a large-scale data fusion task. It is emphasized in [2] that “A significant challenge remains for IDSs designers to combine data and information from numerous heterogeneous agents (and managers) into a coherent process, which can be used to evaluate the security of cyberspace” and that multi-sensor data fusion view of intrusion detection is able to provide an advantageous framework for IDSs of the next generation.

Contemporary view on the problem of information security is concerned with an idea that particular protective mechanisms and corresponding software must be integrated into a distributed system of autonomous software entities interacting via message exchange and making decisions in a cooperative and coordinated manner. These software entities should be adaptive to network traffic variations, to reconfiguration of the network software and hardware components and to unknown types of attacks [3].

The paper is focused on the formal frameworks, architectures and implementations of case studies destined for the exploration of *potential advantages of multi-agent architecture* as applied to different but closely interconnected aspects of computer network security assurance that are (1) Agent-based Simulation of Attacks against Computer Networks (ASACN); (2) Multi-agent Intrusion Detection; and (3) Multi-agent Intrusion Detection Learning.

The chosen strategy of the network security applications implementation was based on the development of specialized software tool that could provide reusability of the most part of software for design a wide range of agent-based network security systems. It resulted in the development of the so-called “Multi-agent System Development Kit” (MASDK). The rest of the paper is structured as follows. Section 2 outlines the basic phases of the supported MASs design technology, generic architecture of agents generated by MASDK and a scheme of interaction of multi-agent systems developed. Section 3 describes the developed attack simulation model and the high-level architecture of the ASACN case study. Section 4 gives an outline of the Multi-agent Intrusion Detection System (MIDS) case study. Section 5 describes the architecture of Multi-agent Intrusion Detection Learning System (MIDLS) and software implementation of its basic components. Section 6 presents a short overview of the existing research relevant to the paper. Section 7 outlines the results and suggests directions of future efforts.

2. DESIGN TECHNOLOGY, AGENT ARCHITECTURE AND SCHEME OF THE DEVELOPED MASs COOPERATION

According to the developed technology the design and implementation of MASs for network security assurance suppose to solve two high level tasks [4]: development of the *System Kernel* of the MAS, and cloning of the software agent instances comprising applied MAS and detachment of the generated MAS from System Kernel.

To design System Kernel, two components of the developed software tool are used. The first of them is so-called *Generic Agent* that aims to support high-level specification of agent classes called at this stage as Generic agent classes. The second component of the software tool is so-called Multi-agent System Development Kit (MAS DK) that is used in specification of the application-oriented components of designed MAS, i.e. architecture, data, knowledge, etc. [4].

The MAS agents generated by MAS DK have the same architecture. Differences are reflected in content of particular agents data and knowledge bases.

Each agent interacts with other agents, environment which is perceived, and, possibly, modified by agents, and user communicating with agents through his interface. *Receiver of input* and *Sender of output* messages perform the respective functions. Messages received are recorded in *Input message buffer*. The order of its processing is managed by *Input message processor*. In addition, this component performs syntax analysis and KQML messages interpretation and extracts the message contents.

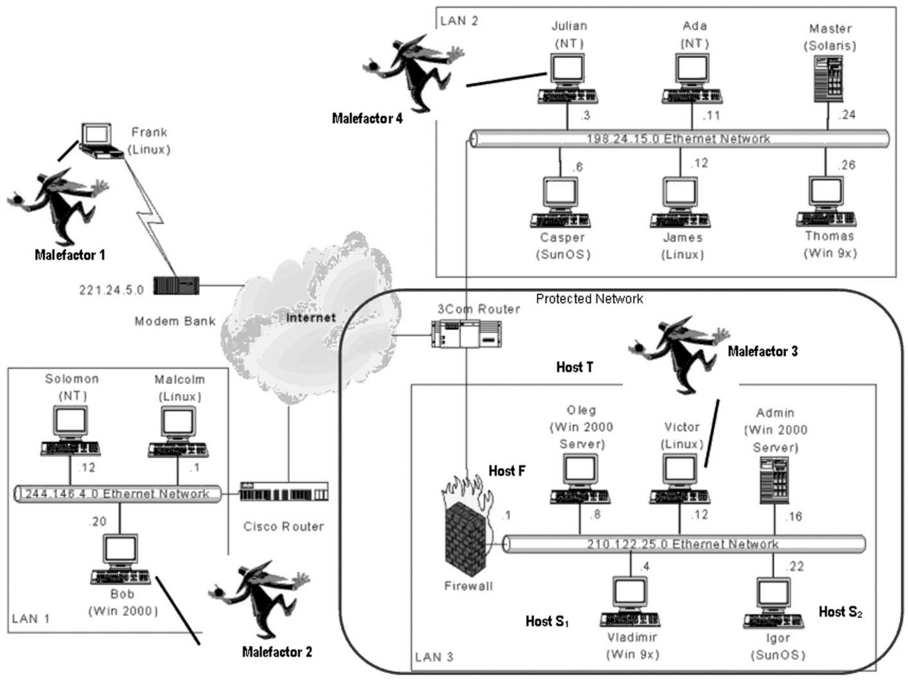
The component *Database of agent’s dialogs* stores for each input message its attributes like identifiers, type of message and its source. If a message supposes to be replied it is mapped the respective output message when it is sent.

Meta-state machine manages the semantic processing of input messages directing it for processing (by) to the respective *State machines*. Another functionality of this component is allocation of management of the parallel performance of agent’s processes. The basic computations of agent corresponding to its role in MAS are executed by a set of *State machines* implemented as automata realizing a scenario of processing of input messages. Each agent class is provided with a set of particular message templates according to its functionalities.

The developer carries out the specialization procedure with *Editor of message templates*, which, in turn, is a component of MAS DK. The message templates are specified in KQML language and specialization corresponds to the assignment to each template the respective performatives.

Communication component of each agent includes also data regarding potential addressees of messages for given template. The last data are assigned at the phase of agent class instances cloning. Message content is specified in XML.

The software code is written on the basis of Visual C++,



ure 2 An example of the network structure

JAVA 2 and XML software development kits. The design and implementation of all three multi-agent case studies for network security assurance is being carried out on the basis of MASDK.

The common scheme of interaction of MASs for network security assurance is depicted in Figure 1.

ASACN simulates the input traffic, i.e. a mixture of normal and abnormal streams of events. The abnormal stream of events sets attacks against the computer network. The input traffic can correspond to a reasonable sequence of these "single-phase" attacks using different entry points (hosts).

MIDS is responsible for detection of attacks against the computer network.

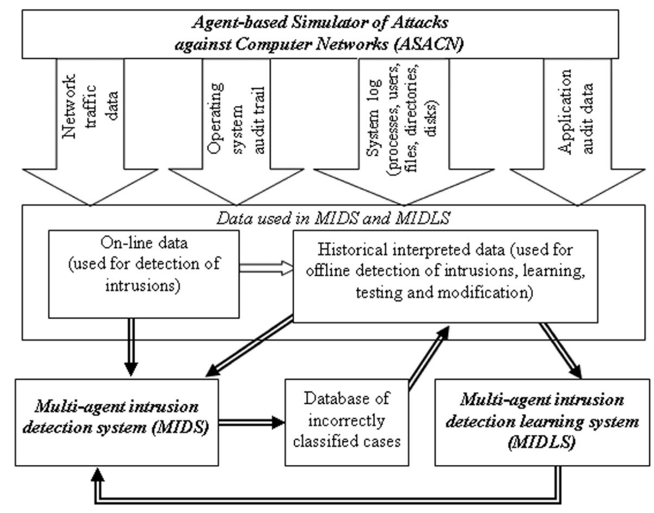

Agent-based Simulator of Attacks against Computer Networks (ASACN)
Network traffic data
Operating system audit trail
System log (processes, users, files, directories, disks)
Application audit data
Data used in MIDS and MIDLS
On-line data (used for detection of intrusions)
Historical interpreted data (used for offline detection of intrusions, learning, testing and modification)
Multi-agent intrusion detection system (MIDS)
Database of incorrectly classified cases
Multi-agent intrusion detection learning system (MIDLS)

Figure 1 Interaction of the input data, intrusion detection and intrusion detection learning systems

MIDLS is designed multi-level learning based on the interpret data from the same s ces and represented in the same structures as the ones used by the MIDS. In the case studies we used different configurations of the network.

This network can comprise several segments of a LAN and the input traffic can be both inside and outside LAN traffic (Figure 2).

3. AGENT-BASED SIMULATOR OF ATTACKS AGAINST COMPUTER NETWORKS

In the developed ASACN [5], distributed attack is specified as a sequence of coordinated actions of distributed malefactors. Each malefactor is mapped as an intelligent agent of the same architecture possessing the similar functionality. While performing a distributed attack, malefactors interact to coordinate their activity.

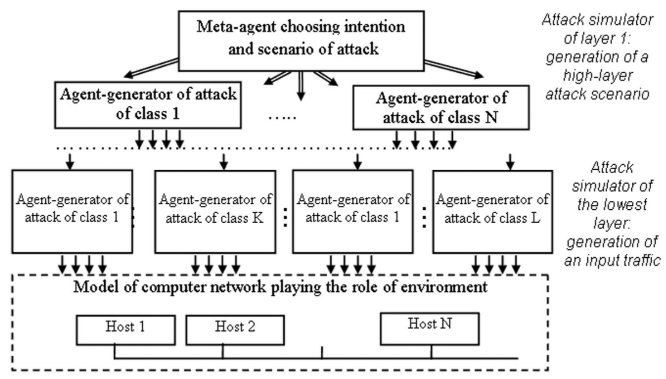

Meta-agent choosing intention and scenario of attack
Attack simulator of layer 1: generation of a high-layer attack scenario
Agent-generator of attack of class 1
.....
Agent-generator of attack of class N
Agent-generator of attack of class 1
Agent-generator of attack of class K
Agent-generator of attack of class 1
Agent-generator of attack of class L
Attack simulator of the lowest layer: generation of an input traffic
Model of computer network playing the role of environment
Host 1
Host 2
Host N

Figure 3 A hierarchical structure of ASACN at implementation of the complex attacks

When implementing the complex coordinated attacks, the special meta-agent forms the common scenario of attack and assigns areas of responsibility to other agents based on the general attack goal installed by user. The agents, responsible for separate fragments (steps) of the common scenario, can in turn "employ" other agents or realize separate operations independently. For this purpose the special scenarios of operations and protocols of messaging are used. The concrete scenario and protocol is (are? termined with usage of the network attacks ontology depending n a type of the realizable goal (intention) the attacked network response. All set of involved nts realizing the concrete scenario compo hierarchical structure (Figure 3).

Each agent (malefactor) of ASACN performs an attack according to a *scenario* represented in high-layer terms. Every phase of the scenario can be detailed in terms of sub-goals of the lower layer. At the lowest one malefactor tries to achieve every upper layer sub-goal through a sequence of acts (commands).

In the developed model we use the *malefactor's intention-centric approach to the specification* of its activity. This means that basic notions of the domain correspond to the malefactor intentions and all other notions are structured accordingly to the structure of intentions.

Attack goal determines the class of scenarios that lead to the intended result. Attack goal is specified by the following quad:

<Network (host) address, Malefactor's intention, Information about attacked network (host) known to malefactor, Attack object>

The basic *malefactor's intentions* are "Resource Enumeration", "Gaining Access to Resources", "Escalating Privilege", etc. *Attack object* corresponds to the optional variable in attack goal specification.

Attack formal model is represented as knowledge base, which is shared by all agents and structured according to the developed domain ontology. The developed ontology includes detailed description of the "Network attack" domain in which the notions of the bottom layer ("terminals") can be specified in terms of audit data and network packets.

Mathematical model of network attacks is specified in terms of the set of stochastic formal grammars interconnected through "*substitution*" operations:

$$M_A = <\{ G_i \} , \{ S_u \}>$$

where $\{G\}$ is the formal grammars, $\{S_u\}$ is the "substitution" operations.

The sequences of symbols generated by each of such grammars correspond to the sequences of time ordered malefactor's intentions and/or actions represented at a layer of details.

Every formal grammar is specified by quintuple

$$G = < V_N, V_T, S, P, A >$$

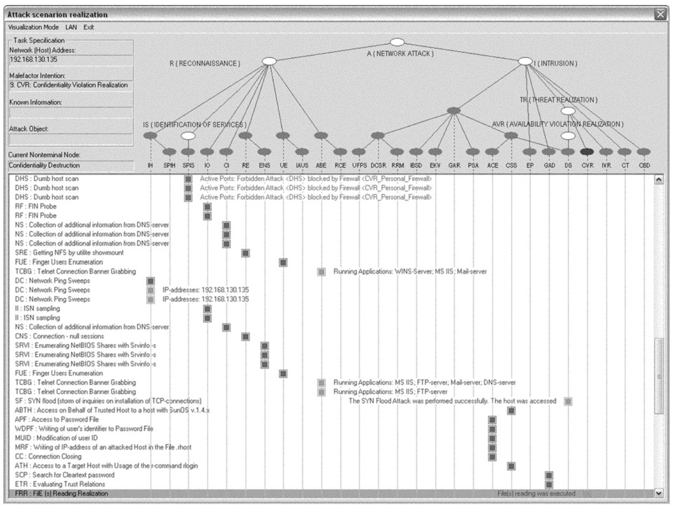
where G is the grammar identifier (name), V_N is the set of non-terminal symbols (that are associated with the upper and the intermediate levels of representation of the steps of an attack scenario), V_T is the set of its terminal symbols (that designate the steps of a lower-level attack), $S \in V_N$ is the grammar axiom (an initial symbol of an attack scenario), P is the set of productions (production rules) that specify the refinement operations for the attack scenario through the substitution of the symbols of an upper level node by the symbols of the lower-level nodes, and A is the set of attributes and algorithms of their computation. The grammar production is recorded as follows:

$$(U)\ X \rightarrow aY\ (Prob)$$

where U is the condition for upholding the rule, X, Y are non-terminal symbols, a terminal symbol, $Prob$ is the probability of the rule bei hosen.

Attribute component of each grammar serves for several purposes. The first of them is to specify *randomised choice of a production* at the current inference step if several productions have the equal left part non-terminals coinciding with the active non-terminal in the current sequence under inference. Also the attribute component is used to check *conditions determining the admissibility of using a production* at the current step of inference.

The family of state machines implements algorithmic interpretation of the attack generation specified as a family


Attack scenarion realization
Visualization Mode
LAN
Exit
A (NETWORK ATTACK)
R (RECONNAISSANCE)
I (INTRUSION)
TR (THREAT REALIZATION)
IS (IDENTIFICATION OF SERVICES)
AVR (AVAILABILITY VIOLATION REALIZATION)
IH SPIH SPIS IO CI RE ENS UE IAUS ABE RCE UFPS DCSR RRM IBSD EKV GAR PSA ACE CSS EP GAD DS CVR IVR CT CBD

Task Specification
Network (Host) Address: 192.168.130.135
Malefactor Intention: 9. CVR: Confidentiality Violation Realization
Known Information:
Attack Object:
Current Nonterminal Node: Confidentiality Destruction

```
DHS : Dumb host scan          Active Ports: Forbidden Attack <DHS> blocked by Firewall <CVR_Personal_Firewall>
DHS : Dumb host scan          Active Ports: Forbidden Attack <DHS> blocked by Firewall <CVR_Personal_Firewall>
DHS : Dumb host scan          Active Ports: Forbidden Attack <DHS> blocked by Firewall <CVR_Personal_Firewall>
RF : FIN Probe
RF : FIN Probe
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
SRE : Getting NFS by utilite showmount
FUE : Finger Users Enumeration
TCBG : Telnet Connection Banner Grabbing          Running Applications: WINS-Server; MS IIS; Mail-server
DC : Network Ping Sweeps
DC : Network Ping Sweeps          IP-addresses: 192.168.130.135
DC : Network Ping Sweeps          IP-addresses: 192.168.130.135
II : ISN sampling
II : ISN sampling
NS : Collection of additional information from DNS-server
CNS : Connection - null sessions
SRVI : Enumerating NetBIOS Shares with Srvinfo -s
SRVI : Enumerating NetBIOS Shares with Srvinfo -s
SRVI : Enumerating NetBIOS Shares with Srvinfo -s
FUE : Finger Users Enumeration
TCBG : Telnet Connection Banner Grabbing          Running Applications: MS IIS; FTP-server; Mail-server; DNS-server
TCBG : Telnet Connection Banner Grabbing          Running Applications: MS IIS; FTP-server
SF : SYN flood (storm of inquiries on installation of TCP-connections)          The SYN Flood Attack was performed successfully. The host was accessed
ABTH : Access on Behalf of Trusted Host to a host with SunOS v.1.4.x
APF : Access to Password File
WDPF : Writing of user's identifier to Password File
MUID : Modification of user ID
MRF : Writing of IP-address of an attacked Host in the File .rhost
CC : Connection Closing
ATH : Access to a Target Host with Usage of the r-command rlogin
SCP : Search for Cleartext password
ETR : Evaluating Trust Relations
FRR : FilE (s) Reading Realization          File(s) reading was executed
```

Figure 4 Visual interface of attack generation

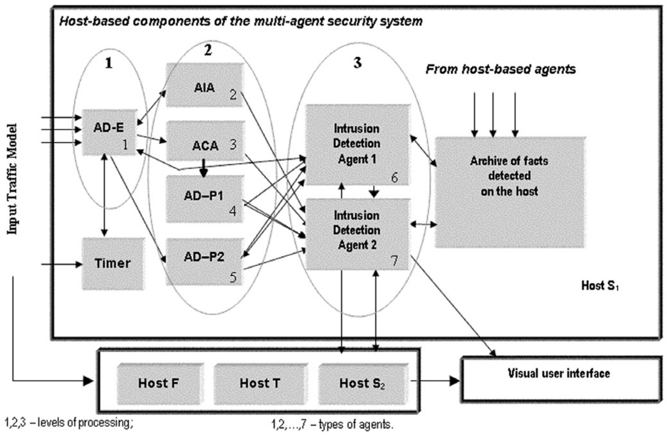

Host-based components of the multi-agent security system
1
2
3
Input Traffic Model
AD-E 1
AIA 2
ACA 3
AD–P1 4
AD–P2 5
Intrusion Detection Agent 1 6
Intrusion Detection Agent 2 7
Timer
From host-based agents
Archive of facts detected on the host
Host S1
Host F
Host T
Host S2
Visual user interface
1,2,3 – levels of processing;
1,2,...,7 – types of agents.

Figure 5 Architecture and interaction of host-based part of MIDS

of formal grammars.

The peculiarity of any attack is that the malefactor's strategy depends on the results of the intermediate actions. This is the reason why the malefactor's action has to be generated on-line in parallel with the getting reaction of the attacked network. The proposed stochastic grammar syntax provides the model with this capability.

The developed version of the software prototype of the ASACN is being used for validation of the accepted ideas, formal framework and implementation issues. The trace fragment for attack "Confidentiality violation realization" generated by ASACN agent is visualized in Figure 4.

4. MUTI-AGENT INTRUSION DETECTION SYSTEM

MIDS makes decisions based on the multi-level input data (network traffic and OS audit data) processing using the meta-classification scheme. The developed case study is an implementation of a particular simplified case of this architecture [3].

The host-based part of the MIDS architecture implemented comprises the following basic components (Figure 5).

Agent-demon AD-E is responsible for input traffic preprocessing. It monitors traffic and extracts sequences of so-called "events" that are semantically meaningful from intrusion detection viewpoint. These events are sorted, stored in *AD-E* database and forwarded to posterior processing to one or several agents.

Agent-demons for identification and authentication (AIA) and *for access control (ACA)* perform their conventional functionalities on identification, authentication and access control, record the results into their data bases and send messages to the *Intrusion detection agent (*IDA*)* if some suspicious behaviour or attempt of an attack has been detected.

Agent-demons AD-P1 and *AD-P2* are responsible for extraction of the "meaningful" patterns of "events" and making decisions regarding to a user's behaviour. In the implemented case study *AD-P1* agent is responsible for extracting patterns associated with the attacks like *finger search* and *buffer overflow*. The agent *AD-P2* is intended to extract patterns corresponding to the *denial of service* attack and *port scanning*.

IDAs make rule-based decisions on the basis of input facts contained in the received messages. They can receive messages about detected suspicious behaviour from agents of the same host as well as other hosts and perform high-level data processing in order to detect multi-phase attacks.

[illegible] brief, three levels of processing to detect an attack can be [illegible]erentiated (see Figure 5):

1. Pre-processing of input traffic by *AD-E* aimed at transformation of the input stream into sequences of significant events;
2. Extracting predefined patterns and detecting the single-phase ("simple") attacks by *AIA, ACA, AD-P1* and *AD-P1*;
3. Detecting combined attacks by *IDA1* and *IDA2*.

The results of testing of the MIDS prototype exhibited the capability of the system to detect multi-phases distributed attacks performed from different source computers (IP-addresses) and through several entry points of the computer network.

[illegible]t us describe an example of some attack detection (Figure [illegible]Vithin simulated attack a malefactor (attacker) target is to get unauthorized access to the resources of the host S_2. The malefactor on the host X is going to act "indirectly", i.e. he uses the access chain S_1 – T – S_2, considering the host S_1 and T as the interim targets.

[illegible]alefactor tries to achieve his targets in seven phases:

1. Scanning of the ports of the host S_1 (successful);
2. Attempts of connection on the host S_1 to access to its ftp or telnet[illegible]vices using various source *IP*[illegible]resses and guessing[illegible]sword (unsuccessful);
3. Realization of the combined spoofing attack (successful);
4. Attempts to get the non-authorized access to the files of the host S_1 (unsuccessful);
5. Buffer overflow attack to boost access rights to the host S_1 resulting in getting access to the password file of the host S_1 (successful);
6. Reading the passwords and getting access to the files of the host S_2 (successful);
7. Connection to the ftp or telnet[illegible]ice of the host S_2 (successful).

[illegible]nput traffic is pre-processed by the agent-demon *AD-E*, [illegible]h re-addresses the resulting messages to the specialized agent-demons *AD-P1* and *AD-P2*. At the first phase of the attack the agent-demon *AD-P2* of the host S_1 is playing the leading role. The second phase involves the agents *AIA* of the same host. The third phase is monitored by the agents *AD-P2* and *IDA1* of the host S_1 and agent *AD-P2* of the host T. During the fourth phase, the agents *AIA* and *ACA* of the host S_1 are operating. Agent *AD-P1* of the host S_1 determines the next phase of attack. During the sixth phase, the agents *ACA* and *IDA2* of the host S_1 are operating. During the final phase of attack the agents *AIA* and *ACA* of the host S_2 are working.

5. MULTI-AGENT INTRUSION DETECTION LEARNING SYSTEM

The main peculiarities and resulting problems of intrusion detection learning technology result from the peculiarities of

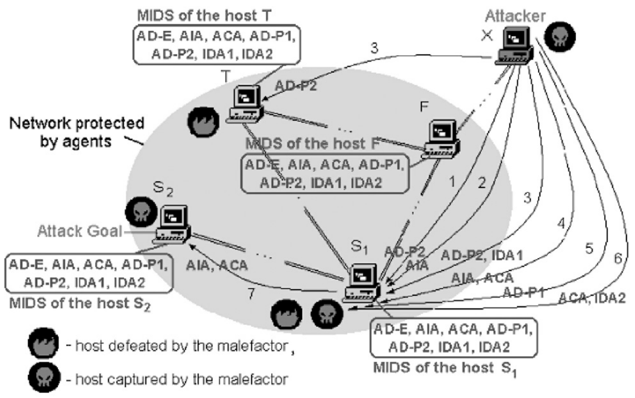

MIDS of the host T
AD-E, AIA, ACA, AD-P1, AD-P2, IDA1, IDA2
T
AD-P2
3
Attacker
X
Network protected by agents
F
MIDS of the host F
AD-E, AIA, ACA, AD-P1, AD-P2, IDA1, IDA2
1
2
3
4
5
6
7
S2
Attack Goal
AD-E, AIA, ACA, AD-P1, AD-P2, IDA1, IDA2
MIDS of the host S2
AIA, ACA
S1
AD-P2
AIA
AD-P2, IDA1
AIA, ACA
AD-P1
ACA, IDA2
AD-E, AIA, ACA, AD-P1, AD-P2, IDA1, IDA2
MIDS of the host S1
- host defeated by the malefactor,
- host captured by the malefactor

Figure 6 Example of attack implementation and their detection by MIDS

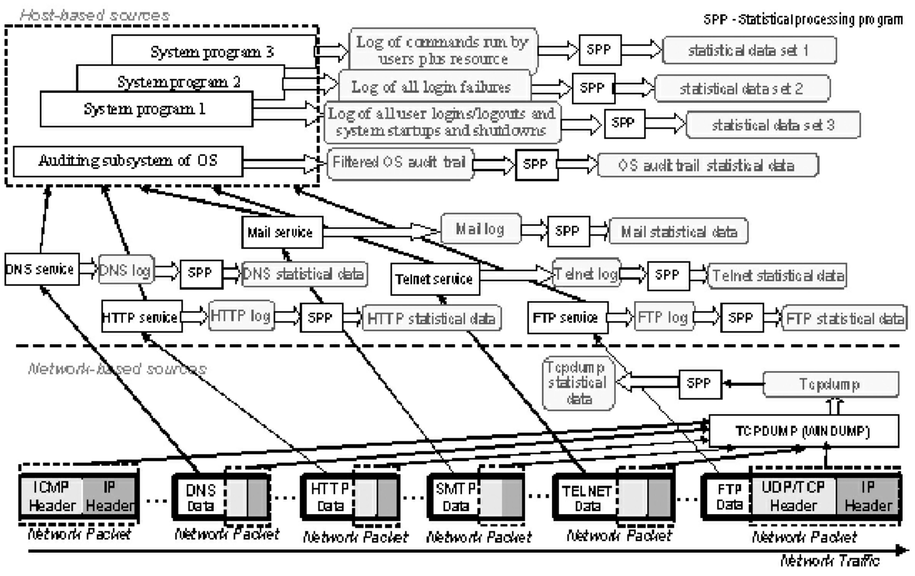


Figure 7 Data Sources used for intrusion detection learning

learning data. The most significant of them results from distributed nature and heterogeneity of data for intrusion detection learning. The traces of illegitimate activity of users (erroneous commands of users, attacks against computer and its information resources) are reflected in multiple distributed and heterogeneous data sources (*tcpdumps*, OS system calls audit trail, application audit data, etc.) (Figure 7). The data can be represented in different data structures (relational sequential, temporal sequential) and measured in different measurement scales (Boolean, categorical, ordered, integer, numerical), be of different accuracy and reliability, they may be incomplete or uncertain and contain missing values, etc.

These properties along with other ones put specific problems within MIDLS design and implementation. These problems, i.e. monosemantic understanding of the terminology used by different components of MIDLS, entity identification problem, problem of diversity of data measurement scales of training data components, "non-coherency of data measurement scales" problem and some others that form together so-called "data non-congruency problem".

The basis of these problem decisions is "ontologism", i.e. approach centered on many-aspect usage of ontology. The technology used supports for the development of shared (common for all software components) and private (available only for particular agents of MIDLS) components of application ontology that are "coherent" with the intrusion detection problem ontology.

Actually three data sources are selected for intrusion detection learning, that are network-based (traffic level), host-based (operating system level) and application-based (FTP server level). It is supposed that each data source is represented by four *generic data structures,* which are the same for each data source. These data structures correspond to the data produced on the basis of processing of raw data. These data structure are as follows:

1. Time ordered sequence of values of binary vectors of parameters specifying significant events of raw data of a level (traffic level, OS logs and FTP server logs). Graphical explanation of this data structures is given in Figure 8. In fact, this data structure specifies time series and for learning techniques developed specifically for discrete vector time series based on correlation and regression analysis is used.
2. Statistical attributes of particular connections (performance of a user) showed in a data source (traffic level, OS logs and FTP server logs). Each such slot of data *Cn:* x_1,x_2, ..., x_k consist of the same attributes for each data source mapped to the time interval of connection appearance (Figure 9).
3. Statistical attributes $X(i) = < x_1, x_2, ..., x_S >$ of traffic (users' activity) during the short term time intervals *d(i)* (2 or 5 second duration). Graphical explanation of this generic data structure is given in Figure 10.
4. Statistical attributes of traffic (users' activity) during the long term time intervals corresponding to a given number of connections (as a rule, *d(i)* = 100 connections that can correspond up to decades of minutes) [6–10] or 10–60 min time interval (Figure 11).

One of the peculiarities of this research is that intrusion detection is considered as an information fusion task in which analysis of current status of users' activity (normal or intrusive) is performed on the basis of distributed heterogeneous data sources. An important peculiarity of such a view

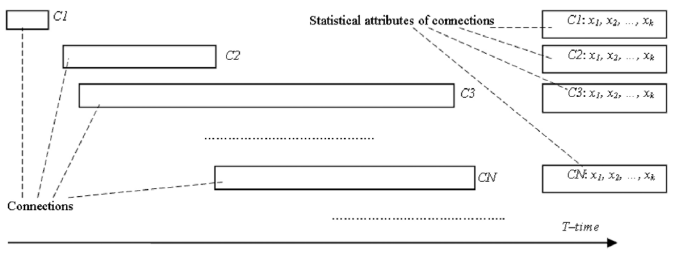


Figure 8 Data structures specifying time-ordered binary vectors of significant events

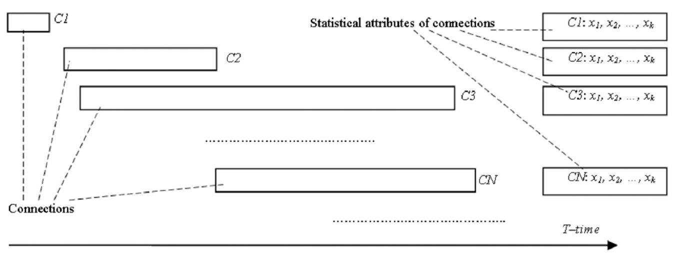

C1
C2
C3
CN
Statistical attributes of connections
C1: $x_1, x_2, \ldots, x_k$
C2: $x_1, x_2, \ldots, x_k$
C3: $x_1, x_2, \ldots, x_k$
CN: $x_1, x_2, \ldots, x_k$
Connections
T–time

Figure 9 Data structures specifying statistical attributes of particular connections

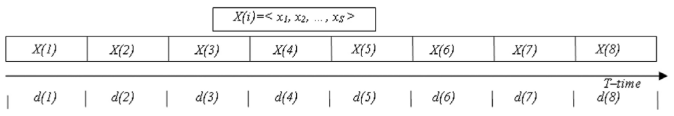

X(i)=< $x_1, x_2, \ldots, x_S$ >
X(1) X(2) X(3) X(4) X(5) X(6) X(7) X(8)
d(1) d(2) d(3) d(4) d(5) d(6) d(7) d(8)
T–time

Figure 10 Data structures specifying integrated statistical attributes of a data source within short term time interval

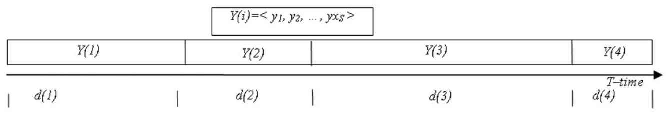


Figure 11 Data structures specifying integrated statistical attributes of a data source within long term time interval

of intrusion detection is that computer network security *situational awareness* results from composition of decisions produced on the basis of particular data providing only *partial awareness*. Thus, intrusion detection is considered as information fusion, respectively, learning of intrusion detection is considered as learning of information fusion.

Multi-agent technology for learning of intrusion detection is a specialization of a technology developed for the design and implementation of more general class of information fusion systems. In this technology applied MAS is designed by use of MAS platform of general purpose followed by use of a toolkit of components called *Information Fusion Learning toolkit*. In this research as MAS platform *Multi-agent System Development Kit*, MASDK, is used [4]. It is noteworthy to note that MASDK and Information Fusion Learning toolkit compose the set of components destined for support of MIDLS design technology and thus together they constitute software tool for the MIDLS technology support.

MIDLS includes several copies of the following classes of agents (Figure 8): learning data management agents; classifier testing agents; meta-data forming agents; and learning agents.

The *learning data management agents* are intended for allocation the training and testing data between different copies of learning agents depending on their roles in the general decision-making structure. In that, *an agent's role is a class of attacks* it is designed to detect and also its *place* in the general decision-making *structure*. At the lower levels, base classifiers make the decisions. There may be several of them for the same subset of attacks, but they should learn on the basis of different sets of training and testing data. At the upper levels, decisions made by the base classifiers are used for making the final decision through combining the decisions made by the base classifiers. This is done by the meta-classifiers. A data management agent possesses information regarding the composition and structure of the entire amount of training and testing data, the composition and the roles of agents in the MIDS within a host. Based on this information

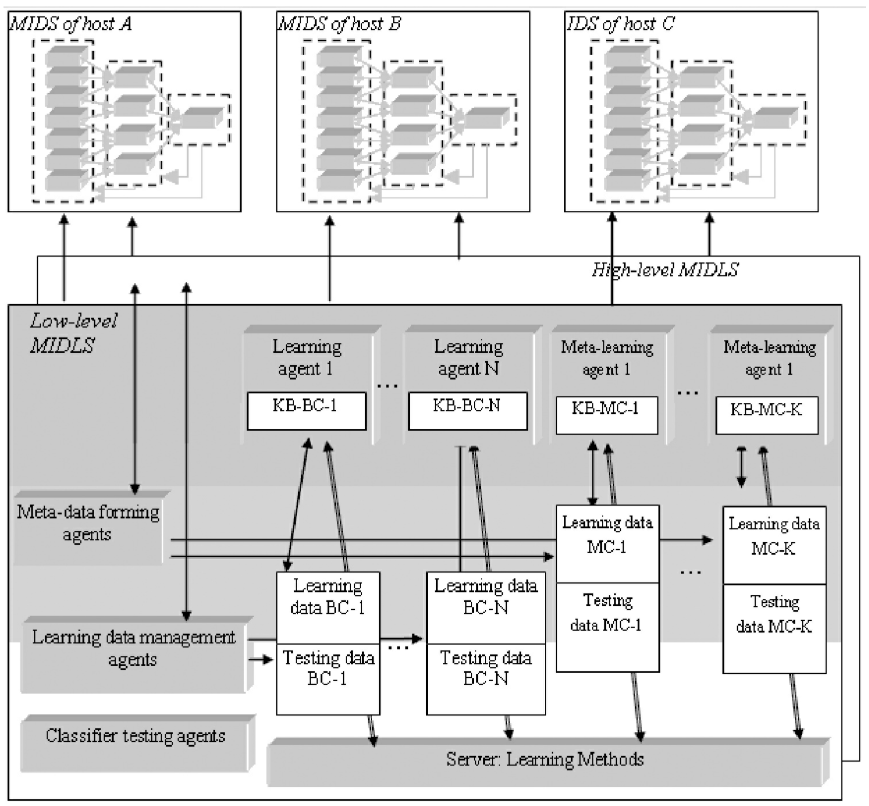


Figure 12 MIDLS architecture

and on its own knowledge, the agent has to propose a decision regarding the partition of the learning sample among different agents of the host.

The responsibility of the *classifier testing agents* consists in the realization of a standard scheme of testing of classifiers based on the data sample chosen as testing and in assessing the learning quality of a classifier based on a specified set of criteria. The second task of these agents consists in launching the "additional learning" procedures if the learning results do not meet the specified criteria. The learning data management agents and the appropriate learning agents are also used in the latter procedure.

The *meta-data forming agents* possess the knowledge concerning to the meta-classifiers for which each of them has to form meta-data for training and testing. This knowledge concerns to the subset of base classifiers, which decisions to be combined. Also each meta-data forming agent has information concerning to the data that serves as the basis for forming the required meta-data sample. This agent's main function consists in launching the testing the required base classifiers and recording the test results in the required format into the database used for meta-data storage.

The *learning agents* realize the main functions of the MIDLS. Two classes of the learning agents are used. The *first class of the learning agents* is designed for the task in which training and testing data are represented as ordered temporal sequences of random length. *The second class of learning agent*s is designed for the learning classifiers that work with the training and testing data represented in the form of attribute vectors. This is the first distinction between the learning agent classes. The second distinction between them lies in the structures of knowledge that they extract from the learning data. Agents of the first class are designed for extracting patterns and frequent episodes. The main functions of the second class of agents consist in extracting rules from the data with attributes represented as vector.

The chosen multitude of learning methods includes both the widely known methods (e.g., ID3, C4.5, boosting, and the meta-classification methodology) and the methods that have been or are being developed by the authors (e.g., visual classification method, the INFORM algorithm, Algebraic Bayes Networks).

6. RELATED WORK

Let us consider the related works in the following order: (1) multi-agent attack simulation tools; (2) multi-agent IDS; (3) multi-agent intrusion detection learning systems.

Till now a lot of data about different security incidents is accumulated. There are a number of publications in which the statistical data of attack cases are systematized in the form of taxonomies (for example, [11–14, etc.]. Nevertheless, there are no serious attempts to generalize the accumulated data in order to develop a *formal model* and simulator of attacks against computer network, in particular, multi-agent model of distributed attacks. The publications reflect a beginning phase of research (see, for example, [15–22]). We developed a strict formal model and techniques for attack modelling based on stochastic formal-grammar-based specification of the scenarios of network attacks on the macro and micro levels.

Recently a number of new models and innovations for IDSs were proposed. Common ideas are: (1) use of knowledge-based frameworks, for example, rule-based systems, neural networks, genetic algorithms, human-like immunology systems; and (2) use of cooperation of the particular distributed components of the network security system that should allow detecting unknown distributed attacks against computer network on a whole. So multi-agent model in IDS design attracted a great attention during several last years [23–25, etc.]). It was ascertained that such a model allows enhancing IDS performance and reliability as compared with conventional approaches. However, the majority of known research using the multi-agent model of IDS and computer network security system on a whole only considers the simplified version of agents' architecture and their cooperative behaviour and do not employ the capabilities of multi-agent technology. In particular, the proposed models and architectures use agents at the pre-processing phase of the protection task. Here the agents are not knowledge-based, are managed by a high-level software manager, restricted by solving intrusion detection task and do not interact with the access control and other security system components. In fact, they do not use a security component cooperation that can provide for a most significant contribution a quality of computer network protection. We implemented the IDS possessing main advantages of intelligent multi-agent systems.

Learning intrusion detection is the problem that is researched about last seven years. The most prominent works on data mining for intrusion detection have been conducted in University of New Mexico (S. Forrest, P. D'haeseleer, S. A. Hofmeyr, etc.), Purdue University (T. Lane and C. E. Brodley), Reliable Software Technologies (A.K. Ghosh, A. Schwartzbard, M. Schatz, etc.), University of Minnesota (V. Kumar, P. Dokas, L. Ertoz, A. Lazarevic, etc.), Columbia University (S. Stolfo, E. Eskin, etc.), North Carolina State University (W. Lee, etc.), Florida Institute of Technology (P. Chan, M. Mahoney, etc.), George Mason University (S. Jajodia, D. Barbara, N. Wu, etc), Arizona State University (Nong Ye, etc.). Of course, this list is not exhaustive.

We can distinguish two main approaches in intrusion detection learning which are shown up more expressively in research of groups of S. Stolfo and S. Forrest.

The basic peculiarity of the approach developing by group of S. Stolfo is that in it intrusion detection learning is considered as a *data analysis process*. Three basic data mining and knowledge discovery algorithms are used for learning intrusion detection classifiers [6–8, 26, 27]. They are the association rules mining, frequent episodes mining and rule extraction algorithms. The first and the second algorithms are used to extract useful associations and serial frequent episodes from historical sequences of audit records, whereas the third one is applied to build knowledge-based classifier. One more specificity of the approach is employment of so-called meta-classification approach [27]. Conceptually, learning system is composed of a number of base classifiers. All decisions of base classifiers are joined in a new record supplemented by the correct interpretation. These records form data for training the meta-classifier. Base classifiers and the meta-classifier interact during solving the intrusion detection tasks in the same way.

An alternative approach to the design and development of the modern learnable intrusion detection systems is being developing in the framework of so-called *"computer immunology"*. The respective results can be found in [28, 29, etc.]. This approach aims to take advantages from the close analogy between the tasks of the computer network intrusion detection system and human immune system. The idea of computer immunology is to use biological principles of human immune system used for distinction between "self" and "non-self" for formalizing processes of intrusion detection in computer network. This approach seems to be very perspective one but now it is at the stage of fundamental research and is exploring usefulness of implementation of some simple basic principles within computer security task.

The approach that is being elaborated by the authors is in some respects close to the approach developed by the group of S.Stolfo. However, this similarity concerns only with the ideas in general. The distinctions are manifold. The *first* of them is in architecture of agents and MAS as a whole. The particular agents are of different specialization. The *second* distinction is that agents are supposed to interact on the basis of shared knowledge represented in ontology, which includes structured problem and subject domain components. The latter makes it possible to deal with detection not only simple attacks against particular computer, but also with detection of distributed attacks against computer network as a whole. The *third* distinction is in techniques for mining knowledge from audit data that we implemented. Together with the standard techniques for learning base classifiers and meta-classifiers we implemented the original ideas and methods. Possibly, the *most important distinction* is that *we consider network intrusion detection task as multi-sensor data fusion.*

7. CONCLUSION

The paper presents the developed multi-agent network security assurance applications implemented by software tool called MAS DK that aims at supporting for the basic phases of MAS technology. These applications made it possible to explore the advantages of using multi-agent technology for computer network security area.

ASACN makes it possible to specify and to simulate distributed attacks at various layers of details using a strict formal model of attack scenario. The variance of attacks is ensured by the random choice of the grammar productions (or the state machine transition rules). The peculiarity of any attack is that the malefactor's strategy depends on the results of the intermediate actions.

The most significant *MIDS* advantage is a capability of comparatively "light" components of a multi-agent security system to cooperate. At present the only way to detect efficiently a distributed attack against a computer network is a cooperation of security agents distributed over the hosts of the network.

MIDLS is viewed as a multi-sensor and a multi-level data fusion system. This system makes decisions on the basis of a multi-level model of network traffic and host-based audit data.

The future research is intended to expand the capabilities of multi-agent network security assurance applications and MAS DK.

ACKNOWLEDGMENT

This research was conducted in Intelligent Systems Laboratory of St. Petersburg Institute for Informatics and Automation. It is being supported by grants 01-01-108 of Russian Foundation of Basic Research and European Office of Aerospace R&D of US AFRL/IF (Projects #1994 P).

REFERENCES

1 Lee, W, Stolfo, S J and Mok, K Mining audit data to build intrusion detection models. *Proc. of 4th International Conference on Knowledge Discovery and Data Mining*, AAAI Press (1998)

2 Bass, T Intrusion Detection System and Multisensor Data Fusion: Creating Cyberspace Situational Awareness. *Comm. of the ACM*, 43 (4) (2000)

3 Gorodetski, V, Kotenko, I, Popyack, L and Skormin, V Integrated Multi-Agent Information Security System: Mechanisms of Agents' Operation and Learning. PAAM' 2000, Manchester. UK. Practical Application Company Ltd. (2000)

4 Gorodetski, V, Karsayev, O, Kotenko, I and Khabalov, A Software Development Kit for Multi-agent Systems Design and Implementation. *Lecture Notes in Artificial Intelligence*, 2296 (2002)

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

6 Lee, W *A Data Mining Framework for Constructing Features and Models for Intrusion Detection Systems*. PhD thesis, Columbia University, New York, NY (1999)

7 Lee, W and Stolfo, S A Framework for Constructing Features and Models for Intrusion Detection Systems. *ACM Trans. on Information and System Security*, 3 (4) (November 2000)

8 Stolfo, S J, Fan, W, Lee, W, Prodromidis, A and Chan, P Cost-based modeling for fraud and intrusion detection: Results from the JAM project. *Proc. of the 2000 DARPA Information Survivability Conference and Exposition (DISCEX '00)*, Hilton Head, SC (2000)

9 Dokas, P, Ertoz, L, Kumar, V, Lazarevic, A., Srivastava, J and Tan, P-N Data Mining for Network Intrusion Detection. *Proc. NSF Workshop on Next Generation Data Mining*, Baltimore, MD (2002)

10 Lazarevic, A, Ertoz, L, Kumar, V, Ozgur, A and Srivastava, J A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection. *3rd SIAConference on Data Mining*, San Francisco, CA (2003)

11 Alessandri, D, Cachin, C, Dacier, M, Deak, O, Julisch, K, Randell, B, Riordan, J, Tscharner, A, Wespi, A and Wüest, C *Towards a Taxonomy of Intrusion Detection Systems and Attacks. MAFTIA deliverable* D3. Version 1.01, Project IST-1999-11583 (2001)

12 Cohen, F B Information System Attacks: A Preliminary Classification Scheme, *Computers and Security*, 6 (1) (1997)

13 Howard, J D and Longstaff, T A *A Common Language for Computer Security Incidents*. SANDIA REPORT, SAND98-8667 (October 1998)

14 Krsul, I V *Software Vulnerability Analysis*. PhD. Dissertation, Computer Sciences Department, Purdue University, Lafayette, IN (May 1998)

15 Chi, S - D, Park, J S, Jung, K - C and Lee, J - S Network Security Modeling and Cyber Attack Simulation Methodology, *Lecture Notes in Computer Science*, 2119 (2001)

16 Eckmann, S T, Vigna, G and Kemmerer, R A STATL: An Attack Language for State-based Intrusion Detection, *Proc. of the ACM Workshop on Intrusion Detection*, Athens, Greece (November 2000)
17 Huang, M - Y and Wicks, T M A Large-scale Distributed Intrusion Detection Framework Based on Attack Strategy Analysis. *Proc. of Second International Workshop on Recent Advances in Intrusion Detection, RAID'98*, Louvain-la-Neuve (1998)
18 Moore, A P, Ellison, R J and Linger, R C *Attack Modeling for Information Security and Survivability*. Technical Note CMU/SEI-2001-TN-001 (March 2001)
19 Moitra, S D and Konda, S L *A Simulation Model for Managing Survivability of Networked Information Systems*[illegible]chnical Report CMU/SEI-2000-TR-020 ESC-TR-2000-020[illegible]cember 2000)
20 Yuill, J, Wu, F, Settle, J, Gong, F, Forno, R, Huang, M, and Asbery, J Intrusion-detection for incident-response, using a military battlefield-intelligence process. *Computer Networks*, 34 (2000)
21 Dawkins, J, Campbell, C and Hale, J Modeling network attacks: Extending the attack tree paradigm. *Workshop on Statistical and Machine Learning Techniques in Computer Intrusion Detection*, Johns Hopkins University (June 2002)
22 Goldman, R P A Stochastic Model for Intrusions. *Lecture Notes in Computer Science*, V.2516. A.Wespi, G.Vigna, L.Deri (Eds.). *Recent Advances in Intrusion Detection. Fifth International Symposium. RAID 2002*. Zurich, Switzerland (2002)
23 Balasubramaniyan, J S, Garcia-Fernandez, J O, Isacoff, D, Spafford, E and Zamboni, D *An Architecture for Intrusion Detection Using Autonomous Agents*.[illegible]ist TR 98-05. West Lafayette, COAST Laboratory, Purdue[illegible]versity (1998)
24 Helmer, G, Wong, J, Honavar, V and Miller, L Intelligent Agents for Intrusion Detection. *Proc. of the IEEE Information Technology Conference, Environment for the Future*. Syracuse (1998)
25 Jansen, W, Mell, P, Karygiannis, T and Marks, D Mobile Agents in Intrusion Detection and Response. *Proc. of the 12th Annual Canadian Information Technology Security Symposium*, Ottawa, Canada (2000)
26 Lee, W, Stolfo, S and Mok, K Algorithms for Mining System Audit Data. *Data Retrieval and Data Mining*. T. Y. Lin and N. Cercone (eds.), Kluwer Academic Publishers (1999)
27 Prodromidis, A, Chan, P and Stolfo, S Meta-Learning in Distributed Data Mining Systems: Issues and Approaches. *Advances in Distributed Data Mining*, Kargupta and Chan (eds.), AAAI Press (1999)
28 Forrest, S, Hofmeyr, S and Somayaji, A Computer Immunology, *Comm. of the ACM*, 40 (10) (1997)
29 Dasgupta, D and Gonzales, F An Intelligent Intrusion Detection System for Intrusion Detection, *Lecture Notes in Computer Science*, 2052 (2001)