Agent-based modelling and simulation of network cyber-attacks and cooperative defence mechanisms

Igor Kotenko
St.-Petersburg Institute for Informatics and Automation of Russian Academy of Sciences
39, 14th Liniya, St. Petersburg, 199178
Russia

1. Introduction

The important *problem in network security* which solution is urgently needed is the investigation of counteraction between malefactors and defence systems in computer networks, including the Internet, and the creation of effective cyber-defence systems.

It is important to underline that experienced malefactors realize *sophisticated strategies of cyber-attacks*. These strategies can include:

- Information gathering about the computer system under attack, detecting its vulnerabilities and defence mechanisms;
- Determining the ways of overcoming defence mechanisms (for example, by simulating these mechanisms);
- Suppression, detour or deceit of protection components (for example, by using slow ("stretched" in time) stealthy probes, separate coordinated operations (attacks) from several sources formed complex multiphase attack, etc.);
- Getting access to resources, escalating privilege, and implementation of thread intended (violation of confidentiality, integrity, availability, etc.) using the vulnerabilities detected;
- Covering tracks of malefactors' presence and creating back doors.

Defence mechanisms should support real-time fulfilment of the following operations:

- Implementing the protection mechanisms appropriated to the security policy (including proactive intrusion prevention and attack blocking, misinformation, concealment, camouflage, etc.);
- Vulnerability assessment, gathering data and analysis of the current status of the computer system defended;
- Intrusion detection and prediction of the malefactors' intentions and actions;
- Direct incident response, including deception of the malefactors, their decoy with the purpose of disclosure and more precise determining the malefactors' purposes, and reinforcement of critical protection mechanisms;
- Elimination of intrusion consequences and detected vulnerabilities, adaptation of the information assurance system to the next intrusions.

The design and implementation of effective cyber-defence system is a very complicated problem. According to contemporary view the *prospective network cyber-defence systems* have to be fully integrated and multi-echeloned ones. To effectively detect computer attacks or unauthorized operations and to flexibly react on them, it is needed to carry out the continuous control of network functioning, analyze possible risks, collect knowledge about counteraction, detection and reaction methods and use them for defence reinforcement.

Besides, the effective cyber-defence should include the mechanisms of attack prevention, detection, source tracing and protection as well as can only be achieved by the cooperation of different distributed components ((Kotenko, 2005), (Kotenko & Ulanov, 2005)).

For example, detection of Distributed Denial of Service (DDoS) flooding attack ((Chen & Song 2005), (Ioannidis&Bellovin, 2002), (Keromytis et al., 2002), (Mirkovic et al., 2002), (Mirkovic et al., 2004), (Mirkovic et al., 2005), (Papadopoulos et al., 2003)] is most accurate close to the victim, but separation of legitimate is most successful close to the sources, therefore the security sub-systems (or teams) have to be located at different network places and tightly cooperate.

The cyber-defence systems have to be adaptive and evolve dynamically with the change of network conditions.

To realize these possibilities in prospective cyber-defence system, one must implement the dynamic behaviour, autonomy and adaptation of particular components, the use of methods based on negotiations and cooperation that lie in the basis of multi-agent systems and (or) autonomic computing.

Furthermore, the prospective cyber-defence system has to provide at least *three levels of cyber-security*.

First level contains *"traditional" static cyber-defence mechanisms* implementing identification and authentication, cryptographic protection, access control, auditing, network filtering, etc.

Second level includes *proactive cyber-defence mechanisms* that provide information collection, security assessment, network state monitoring, attack detection and counteraction, malefactor deception, etc.

Third level corresponds to *cyber-defence management* that fulfils the integral evaluation of network state, the choice of adequate or optimal defence mechanisms and their adaptation. This level is built on top of various non-adaptive security mechanisms, which makes it applicable for a wide range of cyber defences.

The issues of modeling and simulation of network security have been actively researched for more than thirty years. The various formal and informal models of particular protection mechanisms were developed, but practically there are not enough works formalizing complex antagonistic character of network security. Understanding of network security as uniform holistic system is extremely hampered. It depends on great many interactions between different cyber warfare processes and is determined by dynamic character of these processes and different components of computer systems. Especially it is fair in conditions of *the Internet evolution to a free decentralized distributed environment* in which a huge number of cooperating and antagonistic software components (agents) interchange among themselves and with people by large information contents and services. Modeling and simulation of these aspects is supposed to put as a basis of our research. This will allow developing an integrated approach to construction of network security systems which can operate in aggressive antagonistic environment.

Our long-term research goal is to develop a powerful simulation framework and software-hardware environment which can help investigate the Internet attacks and defense mechanisms and elaborate well-grounded recommendations to choose efficient defense mechanisms and develop effective cyber-defence systems.

In our previous papers ((Kotenko, 2005), (Kotenko & Ulanov, 2005), (Kotenko, 2007)) we have examined the common approach to agent-based simulation of network defense mechanisms, types of team-based cooperative defense, and various adaptation schemas.

This paper considers and advances the approach to agent-based simulation of cyber-attacks (Distributed Denial of Service, network worms, botnets, etc.) and distributed cooperative multi-level cyber-defence for the exploration of prospective intelligent cyber-defence systems.

The approach is based on the agent-based simulation of cyber-attacks and cyber-protection mechanisms which combines discrete-event simulation, multi-agent approach and packet-level simulation of network protocols.

We analyze various methods of counteraction against cyber-attacks by representing attack and defence components as agent teams using the simulation environment developed. Various teams of defence agents are able to cooperate as the defence system components of different organizations and Internet service providers (ISPs).

Thus, the paper represents the conceptual framework for modelling and simulation, the implementation peculiarities of the simulation environment as well as the experiments aimed on the investigation of distributed network attacks and defence mechanisms.

The rest of the paper is structured as follows. *Section 2* outlines the common multi-agent modelling and simulation framework suggested and relates work. The classes of agents, used for network attack and defence simulation, and their cooperation schemes are considered in *section 3*. *Section 4* describes the implementation peculiarities of the simulation environment under development. *Section 4* demonstrates the examples of experiments provided with the simulation environment. Conclusion surveys the main results of the paper.

2. Simulation Framework

The *multi-agent approach to simulation* supposes that the cyber-counteraction is represented as the interaction of different teams of software agents ((Kotenko, 2005), (Kotenko & Ulanov, 2005)). The aggregated system behaviour becomes apparent by means of local interactions of particular agents in dynamic environment that is defined by the model of computer network.

Agents of different teams can be in indifference ratio, cooperate or compete up till explicit counteraction. Agents are supposed to collect information from various sources, operate incomplete knowledge, forecast the intentions and actions of other agents, try to deceive the agents of competing teams, react to actions of other agents. Every team member might have different information about actions done by other team members.

Therefore, the model of agent behaviour must be able to represent the incompleteness of information and the possibility of accidental factors. Besides, the agent behaviour depends on information that the team has and on its distribution on the set of particular agents. The models of agent functioning are to foresee, what each agent knows, what task has to be solved and to which agent it must address its request to receive such information, if it is outside of its competence.

The **general conceptual model of cybernetic agents' counteraction and cooperation** includes (Fig. 1) ((Gorodetski & Kotenko, 2005), (Kotenko & Ulanov, 2006)):

- Ontology of application domain containing application notions and relations between them (we differentiate the problem ontology, the shared application ontology, the application ontology of particular team and particular agent);
- Protocols of teamwork for the agents of different teams;
- Models of scenario behaviour of agents for team, group and individual levels;
- Libraries of agent basic functions;
- Communication platform and components for agent message exchange;
- Models of functioning environment, including topological, functional and other components;
- Models that provide the interaction of teams (antagonistic and non-antagonistic competing or various kinds of cooperation).

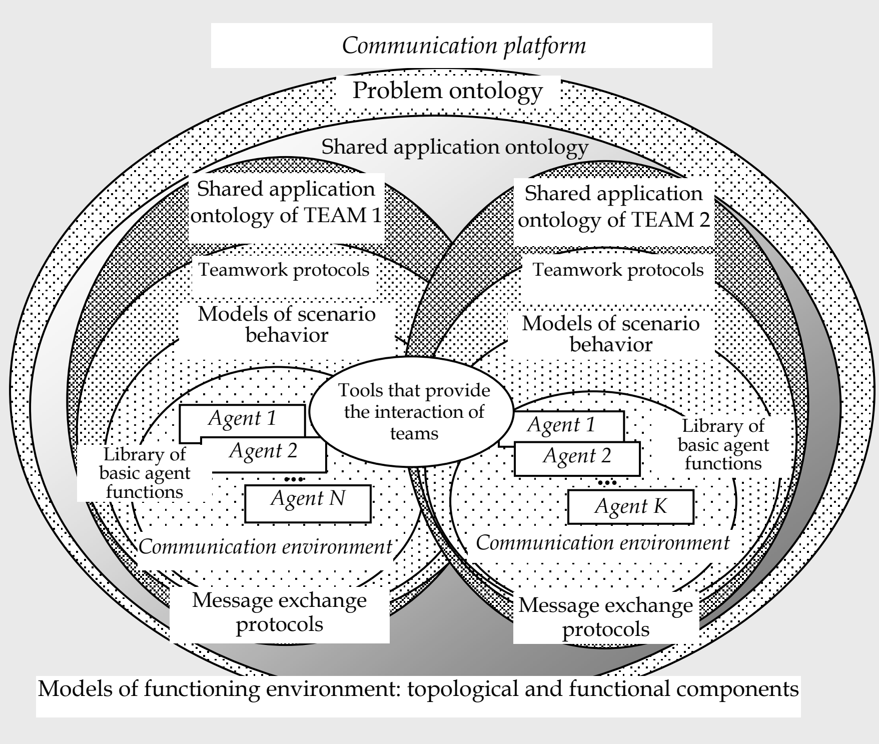

Communication platform
Problem ontology
Shared application ontology
Shared application ontology of TEAM 1
Shared application ontology of TEAM 2
Teamwork protocols
Teamwork protocols
Models of scenario behavior
Models of scenario behavior
Tools that provide the interaction of teams
Agent 1
Agent 2
Agent N
Agent 1
Agent 2
Agent K
Library of basic agent functions
Library of basic agent functions
Communication environment
Communication environment
Message exchange protocols
Message exchange protocols
Models of functioning environment: topological and functional components

Fig. 1. Abstract model of team interaction

The following **main simulation components** are represented on the basis of this approach:

- Models of agent teams;
- Models of team interactions;
- Interaction environment model.

Models of agent teams are intended for the representation of investigated processes. They include: particular team ontologies, agent basic functions, agent classes, agent interaction protocols, behaviour scenarios.

Team ontologies are based on the subject domain ontology and include the notions and relations used by agents of this team.

The list of *agent basic functions* includes the following functions: initialization; shutdown; access to the agent ontology; management of active agents list; basic work with transport-level modules (connection establishing, message sending, connection closing).

The needed *agent classes* are defined for the teams. The amount of agents of predefined classes is set in each team.

Agent interaction protocols are represented as the sequence of instructions with specific parameters. The type of instruction defines how to use these parameters. The conditions of protocol initialization provide communication selectivity for agents. Agent interaction protocols are based on the transport layer that is provided by the communication environment. For example, the developed protocol for agent team establishing is based on dividing agents into “clients” and “servers”. The first send the messages about their existence to “server”. Server contains the list of agents in the team. Periodically it checks the agents in this list to actualize it and to know which of agents are active. Agent team establishing protocol is the part of procedures for monitoring and recovery of agent functionality.

Behaviour scenarios represent various stages of team actions. Adaptation procedures are implemented in scenarios to act depending on other team actions and environment reaction. Agent teams’ behaviour scenarios ensure action consistency maintenance.

Models of team interactions include the models of antagonistic competing, team cooperation and adaptation.

Model of antagonistic competing lies in the basis of competing teams’ interaction. This model defines the goals, subgoals, intentions and actions of competing teams that are aimed on the interaction environment or (and) the opponent team.

Cooperative interaction happens between teams that pursue the same goal. The proposed model of cooperation is based on the exchange of information between teams. Such exchange is made to raise the effectiveness of reaching the common goal and occurs on several different levels with the use of agents of various classes. For example, in the task of cooperative network defence simulation it is possible to exchange attack signatures, network traffic data, filtering requests, etc.

Adaptation is in reacting to the actions of other teams and environment changes by modifying the scenarios of team behaviour.

Model of environment for agent teams’ interaction allows determining such interaction environments that are characterized by various representation granularity that depends on the requirements for simulation fidelity and scalability.

As for every application domain the *ontology* represents the partially normalized set of notions that are to be used by other agents. The ontology defines the subset of notions that various agents use for cooperative solving of stated tasks. Each agent uses a certain part of application domain ontology.

Each agent specialization is represented by the subset of ontology nodes. Some of ontology nodes can be shared by the pair or more of agents. Usually only one of these agents has the detailed description of this node. Exactly this agent is the owner of the corresponding knowledge base fragment. At the same time some part of ontological knowledge base is shared for all agents. This part is the fragment that is to be the shared context (shared knowledge). The structure of agent team is described in terms of group and individual roles hierarchy. The mechanisms of agent interaction and coordination are based on the following procedures: action consistency maintenance; monitoring and the recovery of agent functionality; and communication selectivity ensuring. The specification of action plans hierarchy is made for every role.

The *main basis for the research is the agent teamwork approaches*: joint intentions theory (Cohen & Levesque, 1991), shared plans theory (Grosz & Kraus, 1996) and the hybrid approaches (Tambe, 1997).

It is supposed to use the combination of *methods and models* to form agent teams, to make agent decisions and to coordinate actions between teams and particular agents (Paruchuri et al., 2006):

(1) traditional *BDI-models* which are defined by schemes of agents functioning determined by subject domain dependencies;

(2) methods of distributed optimization on the basis of constraints that use local interactions while searching local or global optimum (*Distributed Constraint Optimization*, DCOP);

(3) methods of distributed decision making on the basis of partly-observable Markov chains that allow to implement the teamwork coordination in the presence of uncertainty in actions and observations (*distributed Partially Observable Markov Decision Problems*, POMDPs);

(4) *game-theoretical models and auction models* focusing on coordination between various agent teams that use market-based decision making mechanisms.

In our approach it is offered that the *agents' teamwork* is organized by the group (team) plan of the agents' actions. In result, a team has a mechanism of decision-making about who will execute particular operations. As in the joint intention theory, the basic elements, allowing the agents' team to fulfil a common task, are common (group) intentions, but its structuring is carried out in the same way as the plans are structured in the shared plans theory. The common (group, individual) intention and commitment are associated with each node of a general hierarchical plan. These intention and commitment manage execution of a general plan, providing necessary flexibility. During functioning each agent should possess the group beliefs concerning other team-mates. For achievement of the common beliefs at formation and disbandment of the common intentions the agents should communicate. All agents' communications are managed by means of common commitments built in the common intentions. For this purpose it is supposed to use the special mechanism for reasoning of agents on communications. Besides it is supposed, that agents communicate only when there can be an inconsistency of their actions. It is important for reaction to unexpected changes of network environment, redistributing roles of the agents which failed or unable to execute the general plan, and also at occurrence of not planned actions.

The *mechanisms of the agents' interaction and coordination* are based on three groups of procedures:

(1) *Coordination of the agents' actions* (for implementation of the coordinated initialization and termination of the common scenario actions);

(2) *Monitoring and restoring the agents' functionality*;

(3) *Communication selectivity support* (for choice of the most "useful" communications).

The specification of the plan hierarchy is carried out for each role. The following elements of the plan should be described: initial conditions, when the plan is offered for fulfilment; conditions for finishing the plan execution (these conditions can be as follows: plan is fulfilled, plan is impracticable or plan is irrelevant); actions fulfilled at the team level as a part of the common plan. For the group plans it is necessary to express joint activity. To cope with the information heterogeneity and distribution of intrusion sources and agents used we apply ontology-based approach and special protocols for specification of shared consistent terminology.

Another fundamental component of the research is represented by the studies on *reasoning systems about opponent intentions and plans* ((Charniak & Goldman, 1993), (Gorodetski & Kotenko, 2005), (Vilain, 1990), (Wellman & Pynadath, 1997)). The important components in this research are the methods of reflexive processes theory (Lefevre, 2003), game theory and control in conflict situations (Druzhinin et al., 1989).

The agents are supposed to implement the mechanisms of *self-adaptation* and to evolve during functioning. The team of agents-malefactors evolves due to generation of new instances and types of attacks and to scenarios of their realization to overcome the defence subsystem. The team of defence agents adapts to malefactors actions due to changing the executed security policy, forming of new defence mechanisms and profiles instances. Therefore it is important to take into account the present studies in the area of adaptation (Silva et al., 2000), agent learning ((Back et al., 2000), (Gamer et al., 2006), (Gu &Yang, 2004), (Zou, et al., 2006)), autonomic computing ((Horn, 2001), (Keromytis et al., 2002), (Want et al., 2003)), and combining artificial immune systems with different computational intelligence methods, such as fuzzy systems, neural networks, etc. ((Ishida, 2004), (Negoita et al., 2005)).

3. Specialised Classes of Agents and their Cooperation Schemes

Let us consider the main classes of teams and agents we currently use in our simulation framework.

There are at least three different *classes of agent teams* (Kotenko & Ulanov, 2006):

- Teams of agents-malefactors,
- Teams of defence agents,
- Teams of agents-users.

Attack agents are subdivided at least into two classes:

- "Demons" and
- "Masters".

Daemons and masters are deployed on compromised hosts in the Internet on a preliminary stage. The class of attacks is defined by intensity of packet sending, IP address spoofing technique (no spoofing, constant, random, and random with real IP addresses), etc.

To simulate distributed cooperative defence, the *security agents* belong to the following classes:

- Information processing ("samplers");
- Attack detection ("detectors");
- Filtering and balancing ("filters");
- Traceback and investigation ("investigators");
- Traffic limiting ("limiters").

Samplers collect and process network data for anomaly and misuse detection. *Detectors* coordinate the team, correlate data from samplers, and detect attacks. *Filters* are responsible for traffic filtering using the rules provided by detector. *Investigator* tries to defeat attack agents. *Limiter* is intended to implement cooperative defence. Its local goal is to limit the traffic according to the team goal. It lowers the traffic to the attack target and allows other agents to counteract the attack more efficiently.

There are three *types of limiting*:

- By the *IP address of attack target*. When detector reveals an attack, it sends to limiter the attack target address. Limiter begins to drop the packets destined to the attack target with the given probability.
- By the *IP addresses of attack sources*. When detector reveals an attack, it sends to limiter the attack source addresses (if detector manages to trace them). Limiter begins to drop the packets from these sources with the given probability.
- According to the *packet marking*. Filter adds to the legitimate-classified packets the mark (it uses one of the packet fields). If limiter sees such mark it does not drop that packet.

Different defence teams can jointly implement investigated defence mechanisms. Defence teams can interact using various schemes. In the one of them that detector acts which team is under attack (when attack is detected). It sends the request to agent-samplers of other teams to receive the information that might be relevant to the mentioned attack. The samplers of other teams reply on the request by sending the requested data. If attack is detected the detector from the victim network sends the information about attack agent addresses to the detector of team in which network this agent might be. Then this team tries to deactivate attack agent. Detector uses the protocol "limiting by the IP address of attack target" to let limiter start or stop the traffic limiting. One of the goals of filter is to add the mark in packets that passed through the filtering table.

The main attention in *cooperative mechanisms* is given to the methods of distributed filtering and rate-limiting. These methods are to trace the attack sources and drop the malicious traffic as far from attack target as possible. The teams of defence agents are able to cooperate as the defence system components of different organizations and Internet service providers.

In the paper we investigate *three cooperative defence mechanisms*:

- DefCOM (Defensive Cooperative Overlay Mesh) (Mirkovic et al., 2005);
- COSSACK (coordinated suppression of simultaneous attacks) (Papadopoulos et al., 2003);
- full cooperation of defence components (suggested in the paper).

The following agent classes are proposed to introduce in compliance with *DefCOM architecture*:

- "Alert generator" – detects an attack and warns about it other hosts in the DefCOM network; attack is detected if the traffic exceeds some threshold;
- "Rate limiter" – limits the traffic that is destined to the attack target;
- "Classifier" – provides selective traffic limiting, tries to classify attack and legitimate packets and to drop the former.

"*Alert generator*" agent is based on "detector" agent. It gathers traffic data from "sampler", detects the IP-addresses of hosts that generate the greatest traffic. If it exceeds the given threshold the alert is generated.

Agent "*Rate limiter*" is based on "limiter" agent. It can drop the packets destined to the attack target providing some volume of traffic.

Agent "*Classifier*" is based on "filter" agent that receives filtering data from detector. This agent is able to filter the disclosed attack packets. It also marks the legitimate packets to let "limiter" pass them. DefCOM "Classifier" receives data from DDoS source network detection system D-Ward) (Mirkovic et al., 2002).
When "Alert generator" detects the attack it sends the attack messages to other agents. Then "Rate limiter" agents start to limit the traffic destined to the attack target. "Classifier" agents start to classify and drop the attack packets and to mark legitimate packets.

COSSACK architecture consists of the following agent classes:

- "Snort" prepares the statistics on the transmitted packets for different traffic flows; the flows are grouped by the address prefix. If one of the flows exceeds the given threshold then its signature is transmitted to "watchdog";
- "Watchdog" receives traffic data from "snort" and applies the filtering rules on the routers.

Agent "*snort*" is based on the agent "sampler". It processes the network packets and creates the model of normal traffic for this network (in the learning mode). Then, in the normal mode, it compares the network traffic with the model and detects the malefactor's IP addresses which it sends to "watchdog".
Agent "*watchdog*" is based on the agent "detector". It makes the decision about attack due to data from "snort".
Agent "filter" is used to simulate filter on the router. It is deployed on router and performs traffic filtering using data from detector.
Detector-level cooperation is used to simulate "watchdog" cooperation. Detectors of different teams are able to transmit the filtering rules to one another due to "filter" agents deployed on routers. Cooperation is in the following: when a "watchdog" detects the attack it composes the attack signature; this "watchdog" sends it to the other known "watchdogs"; "watchdogs" try to trace in their subnets the attack agents that send attack packets; when they detect them the countermeasures are applied.

Full cooperation architecture stipulates for the following classes of defence agents: "samplers", "detectors", "filters", and "investigators".
In common, agent teams are able to interact using various *cooperation schemes*:

- *No cooperation*: all teams work on their own;
- *Filter-level cooperation*: team which network is the attack victim can apply the filtering rules on filters of other teams;
- *Sampler-level cooperation*: team which network is the attack victim can receive traffic data from the samplers of other teams;
- *Poor cooperation*: teams can receive traffic data from the samplers of some other teams and apply the filtering rules on filters of some other teams. Each team "knows" some other teams depending on cooperation degree;
- *Full cooperation*: the team which network is an attack victim can receive traffic data from the samplers of other teams and apply the filtering rules on the filters of other teams.

4. Simulation Environment

A multi-level software environment was supposed to be developed to implement the proposed approach. It differs from the known tools for agent-oriented simulation, as the basis for simulation the tools should be used which provide adequate simulation of network processes.

The spectrum of possible approaches to modelling and simulation are differentiated from analytical to scaled-down and full-scale (see Fig. 2) (Perumalla & Sundaragopalan, 2004).

The choice of model depends on the needed fidelity and scalability of simulation.

The scalability is defined as number of network host (client hosts and routers) which can be simulated using the given method.

The fidelity is defined as a degree of network and hosts destabilization used.

Analytical models allow simulating large-scale Internet processes (including DDoS attacks and worms epidemics) but these models describe the processes only on abstract level.

Packet-level simulation allows enough adequate rendering of such processes. The defence and attack actions are represented as the exchange of packets. This allows the high-fidelity simulation of datalink, network, transport and application layers.

The highest fidelity is reached on the hardware testbeds, but the size of simulated network is restricted enough.

We have chosen the packet-based approach as it provides acceptable scalability and fidelity. In Fig. 2, various program simulators, which can be used for simulation, are depicted: NS2, OMNeT++ INET Framework, SSF Net, J-Sim, etc.

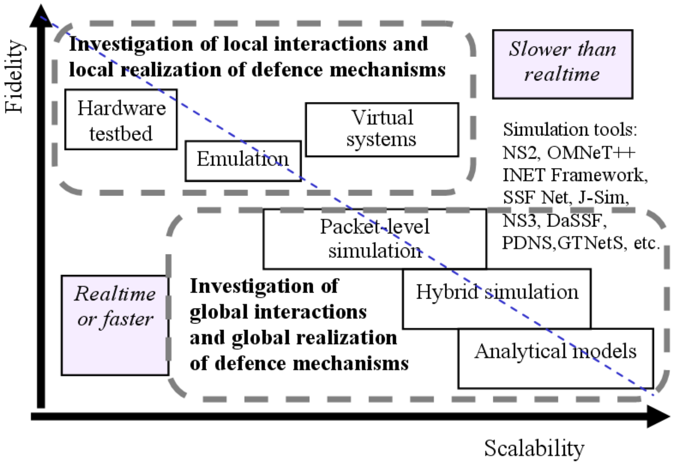

Fidelity
Investigation of local interactions and local realization of defence mechanisms
Slower than realtime
Hardware testbed
Virtual systems
Emulation
Simulation tools: NS2, OMNeT++ INET Framework, SSF Net, J-Sim, NS3, DaSSF, PDNS,GTNetS, etc.
Packet-level simulation
Realtime or faster
Investigation of global interactions and global realization of defence mechanisms
Hybrid simulation
Analytical models
Scalability

Fig. 2. Variety of used models

We take into account the following *main requirements to the simulation environment* (Kotenko & Ulanov, 2005):

- The detailed implementation of the protocols that are engaged in DDoS attacks. It is necessary at least to simulate the present DDoS attacks.
- The ability of writing and plugging in the personal modules. It is necessary to implement the agent approach.

- The ability of changing parameters during the simulation.
- Implementation for OS Windows and Linux (or platform-independency).
- Advanced graphical interface.
- Free for use in research and educational purposes.

To choose the necessary tool we fulfilled the detailed analysis of these simulation environments, and OMNeT++ INET Framework was chosen (OMNeT++, 2010).

The simulation environment architecture suggested includes the following components (Fig. 3):

- Simulation Framework (discrete event simulator),
- Internet Simulation Framework (modular simulation suite with a realistic simulation of Internet nodes and protocols),
- Multi-agent Simulation Framework (modules representing the intelligent agents implemented as application),
- Subject Domain Library (attack and defence modules).

Simulation framework is a discrete event simulator. Other components are expansions or models for Simulation Framework.

Internet Simulation Framework is a modular simulation suite with a realistic simulation of Internet nodes and protocols. The highest IP simulation abstraction level is the network itself, consisting of IP nodes. IP node corresponds to the computer representation of Internet Protocol. IP node can represent router or host. IP node in Internet Simulation Framework corresponds to the computer representation of Internet Protocol. The modules of IP node are organized as operating system process IP datagram. The module that is responsible for the network layer (implementing IP processing) and the "network interface" modules are mandatory. In addition one can plug the modules that implement higher layer protocols.

Multi-agent Simulation Framework allows realizing agent-based simulation. It consists of modules representing the intelligent agents implemented as applications. There were used the elements of abstract FIPA architecture during agent modules design and implementation. Agent communication language is implemented for the agent interactions. The message transmission occurs above the TCP protocol (transport layer) implemented in Internet Simulation Framework. Agent directory is mandatory only for agent that coordinates other agents in its team. Agent can control the other modules due to messages.

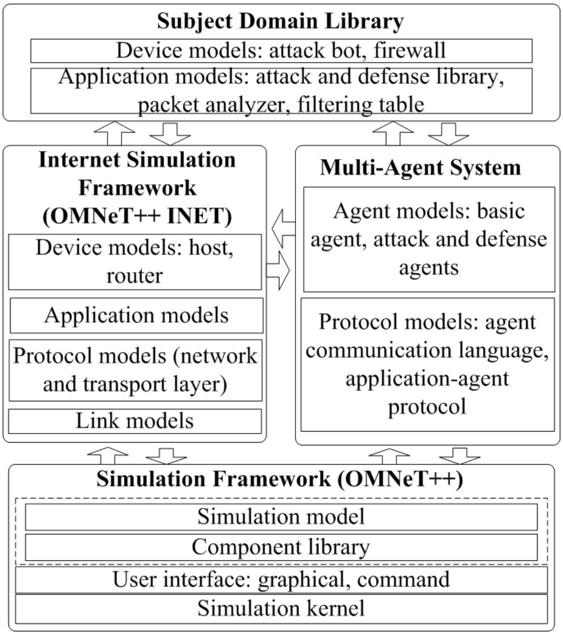
*Subject Domain Libra*ry is the library used for imitation of processes from subject domain and containing modules that extend functionality of IP-host: filtering table and packet analyzer.

This architecture was implemented for multi-agent simulation DDoS attack and defence mechanisms with the use of OMNeT++ INET Framework and software models developed in C++.

Agent models implemented in *Multi-agent Simulation Framework* are represented with generic agent, attack and defence agents.

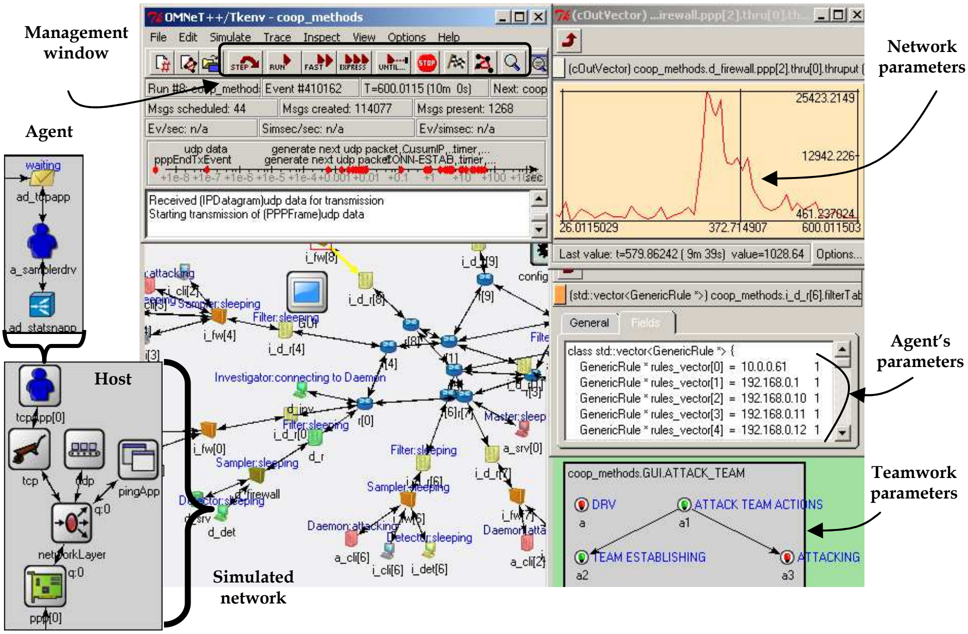
Subject Domain Library contains various models of hosts, e.g. attacking host, firewall etc., and also the application models (attack and defence mechanisms, packet analyzer, filtering table).

Fig. 4 shows the multi-window user interface of the simulation environment.



Subject Domain Library
Device models: attack bot, firewall
Application models: attack and defense library, packet analyzer, filtering table
Internet Simulation Framework (OMNeT++ INET)
Device models: host, router
Application models
Protocol models (network and transport layer)
Link models
Multi-Agent System
Agent models: basic agent, attack and defense agents
Protocol models: agent communication language, application-agent protocol
Simulation Framework (OMNeT++)
Simulation model
Component library
User interface: graphical, command
Simulation kernel

Fig. 3. Simulation environment architecture



```
Received (IPDatagram)udp data for transmission
Starting transmission of (PPPFrame)udp data
```

```
class std::vector<GenericRule *> {
  GenericRule * rules_vector[0] = 10.0.0.61    1
  GenericRule * rules_vector[1] = 192.168.0.1  1
  GenericRule * rules_vector[2] = 192.168.0.10 1
  GenericRule * rules_vector[3] = 192.168.0.11 1
  GenericRule * rules_vector[4] = 192.168.0.12 1
```

Fig. 4. Multi-window user interface of environment

The management window has the time axis with the system events: opening or closing the TCP connection, attack signals, defence acts, etc.

The simulated network window depicts the network hosts and channels. Hosts can fulfil different functionality depending on their parameters or a set of internal modules. Internal modules are responsible for functioning of protocols and applications at various levels of OSI model. Hosts are connected by channels which parameters can be changed. Applications (including agents) are established on hosts. Applications are connected to corresponding modules of protocols.

The structure of generic host is depicted on the bottom left. The deployed agent is represented as the blue symbol of human in the frame.

The environment allows to examine the different information describing the simulation functioning. For example, the diagram that shows the change in network parameters is depicted at the top right.

The networks used for simulation consist of various subnets that are, for instance, the regions of responsibility of various ISPs.

For example, one can mark out the defence subnet where the attack victim is located, the intermediate subnets where the standard hosts generate generic network traffic, and attack subnets where the attack agents are located.

The networks are built with the methods of generating topologies that are close to the real Internet (Mahadevan et al., 2005).

There are the following *specification elements to define the investigated network models, attack and defence mechanisms*:

- Network topology: quantity and types of hosts, channels between them and their types. The possibility to deploy certain type of application (or agent) depends on host type.
- Defence team parameters: quantity of daemons; master's address and port used for interactions; daemon's port used to send attack packets; victim's address and port; time of attack; attack intensity; address spoofing technique.
- Attack realization parameters: victim type (application, host or network; one must define the IP-address and port of victim); type of attack (brute force (UDP/ICMP flood, smurf/fraggle, etc.) or semantic (TCP SYN, incorrect pack-ets, hard requests, etc.)); attack rate dynamics (can be constant or variable); adaptation scheme depending on attack severity, etc.
- Defence team parameters: address of defended host; detector's address and port for interactions; server's reply size and delay time; adaptation scheme (changing of defence mechanisms) depending on attack severity, etc.
- Defence mechanisms parameters: deployment location (source, intermediate or defended subnets); the stages the defence method can implement (attack prevention, attack detection, tracing the attack source, attack counteraction); attack detection technique (misuse and anomaly detection; one chooses one particular detection method or the set of methods), etc.
- User team parameters: quantity of users; server's address and port; time to start; quantity of requests to server, interval between them and their size; interval between connections.
- Defence team cooperation parameters: scheme of cooperation.
- Simulation parameters: simulation duration; quantity of experiments; initialization of random number generator.

5. Experiments

The developed simulation environment allows carrying various experiments aimed to investigate attacks and prospective defence strategies. One can vary network topology and configuration, structure and configuration of attack and defence teams, attack and defence mechanisms, team cooperation parameters etc. The evaluations of various effectiveness parameters of defence mechanisms are done on the basis of experiments results. The analysis of applying conditions is also fulfilled for these parameters.

The attack parameters used in the experiments are as follows: Victim type – host (server that provides some service); Attack type – brute-force; Impact on the victim – disruptive; Attack rate dynamics – constant, variable; Agents' set permanency – constant, variable; Possibility of exposure – discoverable filterable attack; Source addresses validity – valid (real), spoofed: random, subnet; Degree of automation – semi-automatic with direct communication.

In the experiments we have used three defence methods: Hop counts Filtering (HCF) (Jin et al.,2003), Source IP address monitoring (SIPM) (Peng et al., 2003) и Bit Per Second (BPS). HCF consists in building the tables of subnets and amount of hops till them in the learning mode. Attack is found out on the basis of amount of hops differing from received in learning mode. SIPM uses the assumption that during attack a lot of new IP addresses appear. BPS allows detecting the attacker due to exceeding the normal traffic threshold.

The other defence parameters are as follows: Deployment location – intermediate, defended subnets; Covered defence stages – attack prevention, attack detection, attack source detection, attack counteraction; Attack source detection technique – can detect when source address is not spoofed; Attack prevention technique – packet filtering; Technique for gathering of model data – learning; Determination of deviation from model data: thresholds (HCF, BPS), determination of fluctuation in probabilistic traffic parameter (SIPM).

Let us consider three examples of experiments fulfilled where we analyzed different modes of DDoS attacks and defence: (1) experiment 1 - investigation of simple adaptation of attack and defence teams; (2) experiment 2 - investigation of different, suggested in the paper, cooperation modes between defence teams; (3) experiment 3 - investigation of cooperative defence mechanisms DefCOM, COSSACK and full cooperation.

Let us consider the values of the main parameters that define the computer network models, attack and defence mechanisms that are used for the experiments.

To create a topology for testing, we used the generator of networks that are close to the real Internet networks. The following basic network topology parameters were set: minimum amount of connection is 2, the amount of routes in simulated networks is 10, the probabilistic value γ = 2.25 (Mahadevan et al., 2005)]. Routers are connected with the fiber-glass data channels, propagation delay is 1 microsec; datarate is 2488 Mbit. Other hosts are connected by Ethernet data channels, propagation delay is 0.1 microsec; datarate is 100 Mbit. Clients are randomly connected to the routers of the basic network. The amount of clients is an input parameter for experiments (its initial value is 10). The defended server is d_srv. The basic parameters of network clients are as follows: server address "d_srv"; server port – 80; start time is a random value with the exponential probability distribution function (PDF) and mean 5 sec; one request per session is used; request length is a random value with normal PDF with mean 350 and dispersion 20 bits; reply length is a random value with exponential PDF and mean 2000 bits; Think time is a random value with normal PDF with mean 2 and dispersion 3 sec; Idle interval is a random value with normal PDF with mean 36 and dispersion 12 sec; Reconnect interval is 30 sec.

Let us consider below the results of experiments fulfilled.

5.1 Investigation of simple adaptation of attack and defense teams

The fragment of the network which was used in the first experiment is depicted in Fig. 5. The fragment of decision making and acting sequence is as follows (Fig. 6):

- Normal work of users (interval 0 – 300 seconds).
- Defence team: formation of the team; the team start using BPS method.
- Attack team: formation of the team; after 300 seconds the team begins the attack actions (intensity of attack for every daemon - 0.5, no IP spoofing).
- Defence team: data processing, attack detecting (using BPS) and reacting (interval 300 – 350 seconds); blocking the attack, destroying some attack agents (interval 300 – 600 seconds).
- Attack team: after 600 seconds the automatic adaptation is fulfilled (redistributing the intensity of attack (0.83), changing the method of IP spoofing (Random)).
- Defence team: data processing, failing to detect the attack (using BPS method) – Detector sees that the input channel throughput has noticeably lowered, but does not receive any anomaly report from sampler because BPS does not work.
- Defence team: Changing defence method on SIPM (automatic adaptation); Data processing, attack detecting (using SIPM method) and reacting – (interval 600 – 700 seconds).

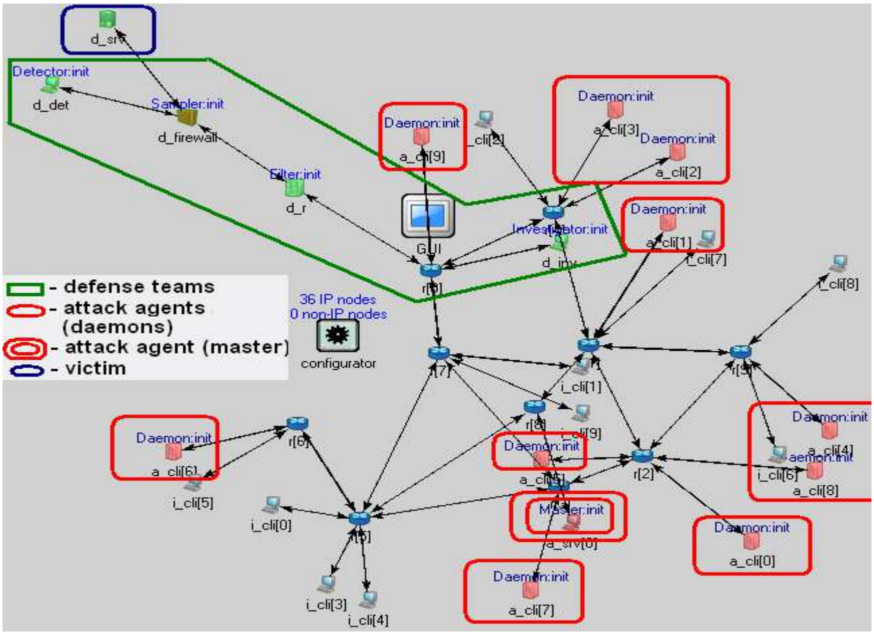


Fig. 5. Experiment 1: the Internet fragment and agent teams

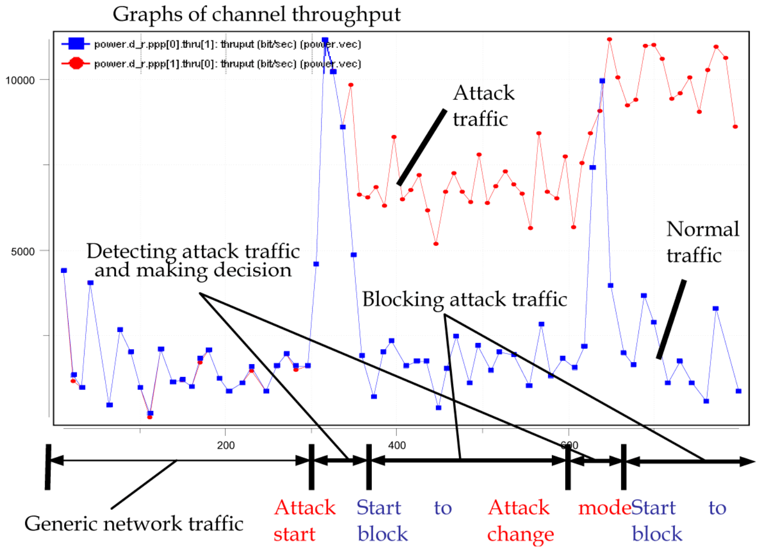


Fig. 6. Scheme of teams' acting

5.2 Investigation of different, suggested in the paper, cooperation modes between defense teams

The fragment of the network which was used in the second experiment is depicted in Fig. 7. We investigated the models of cooperation between distributed defence teams:

(1) filter-level cooperation: the team whose network is under attack can apply filtering rules on the filters of other teams;

(2) sampler-level cooperation: the team whose network is under attack can get the traffic information from the samplers of other teams;

(3) "poor" cooperation: the teams can get the traffic information from the samplers of some other teams and apply filtering rules on the filters of some other teams (each team knows a subset of other teams depending on the cooperation degree);

(4) "full" cooperation: the team whose network is under attack can get the traffic information from all samplers of other teams and apply filtering rules on all filters of other teams.

Fig. 8 depicts the volume of input traffic before and after the filter of the team which network is under attack when the BPS method is used.

The other effectiveness and efficiency parameters of different defence mechanisms which were investigated are as follows: rate of dropped legitimate traffic (false positive rate); rate of admitted attack traffic (false positive rate); attack reaction time.

These parameters were investigated in dependence on the following input parameters: network configuration; attack intensity; IP address spoofing technique used in attack; internal parameters of defence mechanisms and their combinations; quantity and distribution of defence teams, etc.

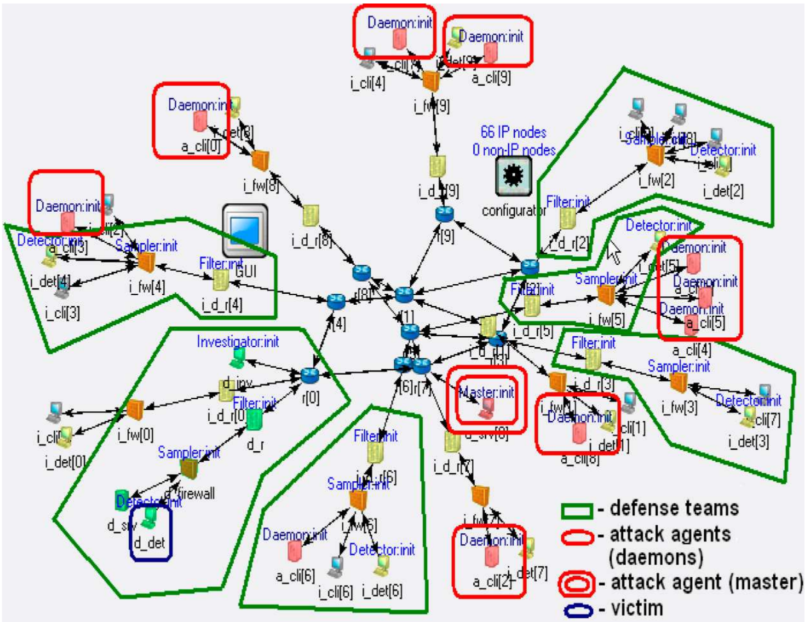
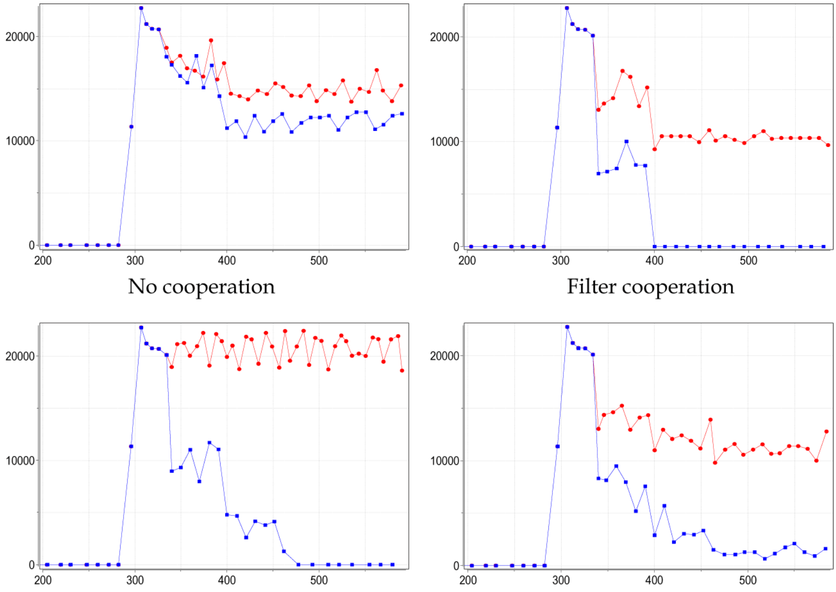


Fig. 7. Experiment 2: the Internet fragment and agent teams



No cooperation

Filter cooperation

Sampler cooperation

Full cooperation

Fig. 8. Volume of input traffic before and after the filter

The best cooperative schema on the basis of output parameters is "full cooperation". Samplers-agents cooperation played the crucial role in defence. It causes the permanent traffic data exchange between various defence teams.

5.3 Investigation of cooperative defense mechanisms DefCOM, COSSACK and full cooperation

Fig. 9 shows the example of DefCOM agents' configuration in the tool developed. Agents "Filter" play the role of "Classifier", "Limiter" – "Rate limiter" and "Sampler" with "Detector" – "Alert generator". Defence team of d_srv host has the following configuration: detector is deployed on the host d_det (in the defended subnet), sampler – d_firewall (on the entrance to the defended subnet), two agents "filter" – on the hosts i_d_r[0] and i_d_r[1] (in the source subnets), limiter – r[0] (router that provides the connection to the defended subnet). Other basic parameters are as follows: detectors port for team interaction – 2000; interval for SIPM and BPS – 5 seconds; time shift for SIPM and BPS – 5 seconds.

Fig. 10 shows the example of COSSACK agents configuration in the tool developed. Defence network consists of three agents "watchdog" that are simulated by the defence agent teams.

Defence team of d_srv host has the following configuration: detector is deployed on the host d_det (in the defended subnet), sampler – d_firewall (on the entrance to the defended subnet), agent "filter" – on the hosts d_r (on the entrance to the defended subnet, before sampler), limiter – r[0] (router that provides the connection to the defended subnet). Two other teams consist of detector and filter in the source subnets (hosts i_cli[1] and i_cli[2], and i_d_r[0] and i_d_r[1] accordingly).

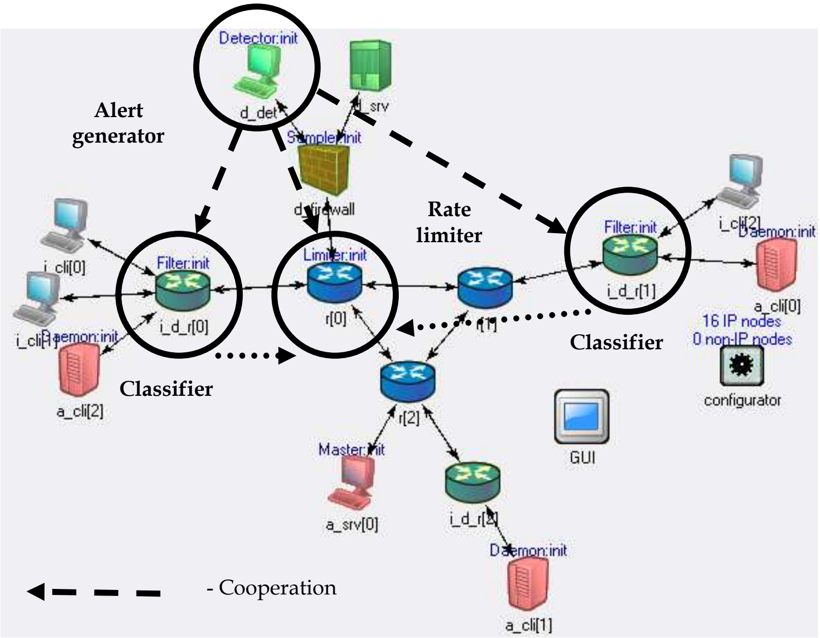


Fig. 9. Configuration of DefCOM agents

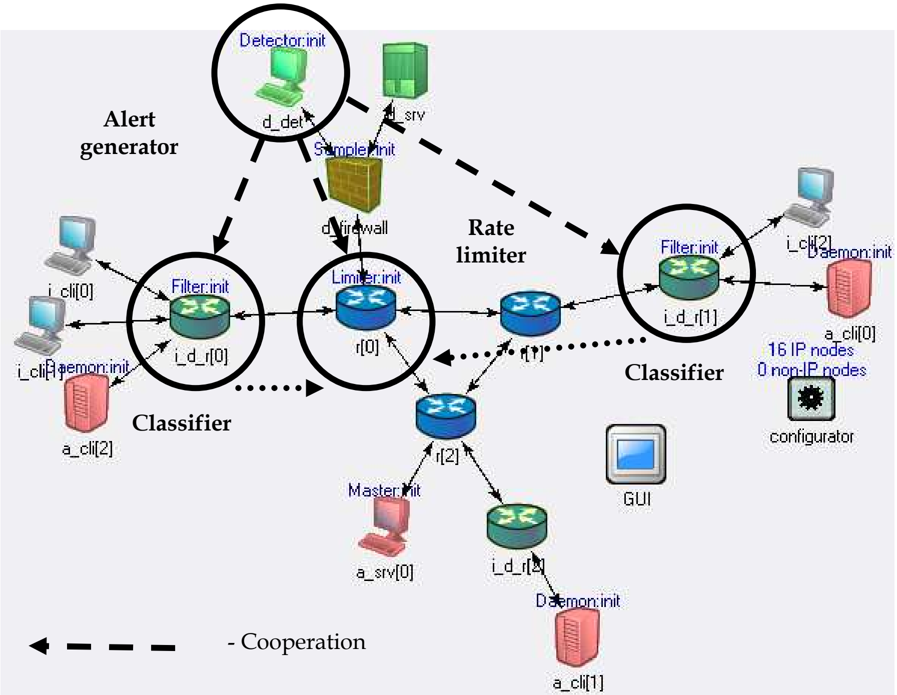


Fig. 10. Configuration of COSSACK agents

Fig. 11 shows the example of Full cooperation defence system proposed by the authors. This defence network consists of defence teams that are able to cooperate to reach the mutual goal.

Each team has the following configuration: detector is deployed on the host d_det[i] (in the defended subnet), sampler – d_firewall (on the entrance to the defended subnet), agent "filter" – on the hosts d_r[i] (on the entrance to the defended subnet, before sampler), investigator is deployed out of defended subnet ("i" is the subnet number).

The following cooperation schemes have been investigated in a set of experiments:

- DefCOM: when an attack is detected "Alert generator" sends the attack messages to the other agents. "Rate limiter" agents start to limit the traffic destined to the attack target. Agents "Classifier" classifies, drops the attack packets, and marks the legitimate packets.
- COSSACK (or filter-level cooperation): the team which network is the attack victim can apply the filtering rules on filters of other teams. When "watchdog" detects the attack it creates attack signature and sends it to the other known "watchdogs"; they try to trace in their subnets the attack agents that send attack packets; when they detect them the packet filtering rules are applied as close as possible to the attack source.
- full cooperation: the team which network is the attack victim can receive traffic data from the samplers of other teams and apply the filtering rules on filters of other teams; this scheme has the best effectiveness comparing with the other proposed.

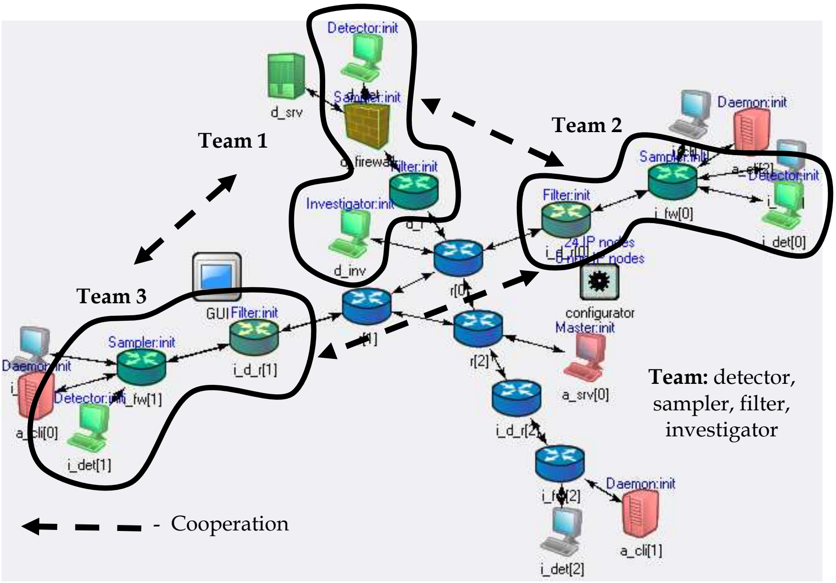


Fig. 11. Configuration of full cooperation defence system

The investigation has been done on the basis of analysis of two main classes of parameters:

- the amount of incoming attack traffic before and after filter of team which network is the attack victim;
- false positive and false negative rates of defence team which network is the attack victim.

Fig. 12 shows the examples of attack traffic inside the attacked subnet for the COSSACK (triangles), DefCOM (dots) and full cooperation schema (crosses). Attack starts at 300 seconds. The random real IP spoofing technique is applied as the most complicated for detection (the addresses for spoofing are taken from the same network). SIPM is used as the defence method.

Attack traffic for COSSACK is measured on the entrance to the defended subnet on filter. The significant traffic increase is noticed in the beginning of attack. But in the area of 350 seconds the defence system detects the attack. Filtering rules are applied and the traffic inside the subnet is reduced (after 350 seconds). Attack signature is sent to the other defence components. They apply filtering rules in their subnets. The traffic on the entrance to the defended subnet is decreased due to their actions.

The attack traffic inside the attacked subnet for DefCOM is represented with the dots in Fig. 12. The traffic was measured at the entrance to the subnet, since the last component in the subnet that changes the traffic is the limiter. It is deployed on the router that has four interfaces and the incoming attack traffic was summarized into one graph. In the area of 350 seconds the defence system detects the attack and traffic is being limited before the defended subnet and being filtered in the source subnets. Rate limiter proceeds to limit the traffic because of the high attack traffic volume. But this cooperation schema succeeds to keep the traffic on the acceptable level due to limiting and to applying of filtering rules.

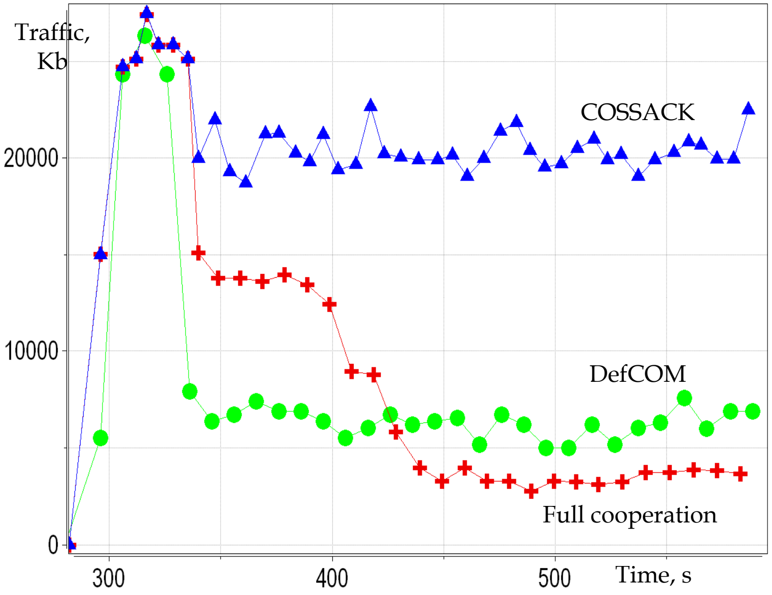


Fig. 12. Attack traffic inside the attacked subnet for COSSACK, DefCOM and full cooperation

The attack traffic inside the attacked subnet for the full cooperation schema is represented with the crosses in Fig. 12. Traffic is measured on the entrance to the defended subnet on filter. The significant traffic increase is noticed in the beginning of attack. But in the area of 350 seconds the main defence team detects the attack requesting the traffic data not only from its sampler but from the samplers of other teams. Filtering rules are applied and traffic inside the defended subnet is significantly decreased (around 350 seconds). Attack signature is sent to the other cooperating teams. They apply the filtering rules in their subnets. The traffic on the entrance to the defended subnet is decreased due to their actions (after 350 seconds). System succeeds to decrease the traffic much more due to permanent attack signatures renewing (400–450 seconds).

The experiments implemented demonstrated that full cooperation shows the best results on blocking the attack traffic. It uses several defence teams with cooperation on the level of filters and samplers.

DefCOM comes after full cooperation. Its advantage is in using the rate limiter before the defended network. It allows lowering the traffic during attack and letting the defended system work properly.

COSSACK is the third. It is one of the examples of peer-to-peer defence network. It uses attack signatures transmission between agents to apply the filtering rules near the source. The communication overhead for cooperative defence is restricted by the communication selectivity procedures. The agent protocols can be executed only periodically or in the strict sequence. Therefore their influence on the joint traffic is low.

6. Conclusion

This paper considered the approach to investigation of distributed cooperative cyber-defence mechanisms against network attacks. The approach is based on the simulation of network cyber-attacks (Distributed Denial of Service, network worms, botnets, etc.) and cyber-protection mechanisms which combines discrete-event simulation, multi-agent approach and packet-level simulation of network protocols.

The environment developed is written in C++ and OMNeT++. It allows imitating a wide spectrum of real life infrastructure attacks and defence mechanisms.

A lot of different experiments were carried. They were aimed to investigate dependence of defence effectiveness parameters from network topology and configuration, structure and configuration of attack and defence teams, attack and defence mechanisms and defence teams' cooperation.

Experiments showed that team cooperation leads to the essential defence effectiveness improvement. The multitude of experiments we implemented demonstrated that full cooperation shows the best results on blocking the attack traffic. It uses several defence teams with cooperation on the level of filters and samplers.

Future work is related with more thorough investigation of effectiveness of cooperation mechanisms for different teams and inter-team interaction of agents, implementation of self-adaptation and self-learning of agents. We are planning to expand the attacks and defences library, elaborate particular components functionalities.

One of the main tasks of our current and future research is to improve the scalability and fidelity of the simulation. We now in the process of designing and experimenting with the parallel versions of our simulation environment and developing a simulation tesbed combining a hierarchy of macro and micro level models of attack and defence (analytical, packet-based, emulation-based), and real small-sized networks.

7. Acknowledgement

This research is being supported by grant of Russian Foundation of Basic Research (Project No. 10-01-00826-a), program of fundamental research of the Department for Informational Technologies and Computation Systems of the Russian Academy of Sciences (contract No. 3.2), Russian Science Support Foundation and partly funded by the EU as part of the MASSIF project (contract No. 257475). Author thanks Aleksey Alekseev, Alexey Konovalov, Alexander Ulanov, and Andrey Shorov for software development and testing.

8. References

Back, T.; Fogel, D.B. & Michalewicz, Z. (2000). *Evolutionary computation*. Vol. 1. Basic algorithms and operators, Institute of Physics Publishing.

Charniak, E. & Goldman, R.P. (1993). A Bayesian Model of Plan recognition. *Artificial Intelligence*, Vol. 64, No. 1.

Chen, S. & Song, Q. (2005). Perimeter-Based Defence against High Bandwidth DDoS Attacks. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 16, No. 7.

Cohen, P. & Levesque, H.J. (1991). Teamwork. *Nous*, No. 35.

Druzhinin, V.V.; Kontorov, D.S. & Kontorov, M.D. (1989). *Introduction into conflict theory. Moscow*, Radio i svyas' (in Russian).

Gamer, T.; Scholler, M. & Bless, R. (2006). A Granularity-adaptive System for in-Network Attack Detection, *Proceedings of the IEEE/IST Workshop on Monitoring, Attack Detection and Mitigation*.

Geib, C.W. & Goldman, R.P. (2001). Plan recognition in intrusion detection systems, *DARPA Information Survivability Conference and Exposition, DARPA and the IEEE Computer Society*.

Gorodetski, V. & Kotenko, I. (2005). Conceptual foundations of stochastic simulation in the Internet. *Proceedings of system analysis institute of RAS*, Vol.9, Moscow, URSS (in Russian).

Grosz, B. & Kraus, S. (1996). Collaborative Plans for Complex Group Actions. *Artificial Intelligence*, Vol. 86.

Gu, D. & Yang, E. (2004). Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey, *Technical Report of the Department of Computer Science*, University of Essex, CSM-404.

Horn, P. (2001). *Autonomic Computing: IBM's Perspective on the State of Information Technology*, Technical Report, IBM Corporation.

Ioannidis, J. & Bellovin, S.M. (2002). Implementing Pushback: Router-Based Defence Against DDoS Attacks, *Proceedings of Symposium of Network and Distributed Systems Security (NDSS)*, California.

Ishida, Y. (2004). *Immunity-Based Systems A Design Perspective*. Springer Verlag.

Jin, C., Wang, H. & Shin, K.G. (2003). Hop-count filtering: An effective defence against spoofed DDoS traffic, *Proceedings of ACM Conference on Computer and Communications Security*.

Kephart, J.O. & Chess, D.M. (2003). The Vision of Autonomic Computing. *IEEE Computer Magazine*, No. 1.

Keromytis, A.; Misra, V. & Rubenstein, D. (2002). SOS: Secure Overlay Services, *Proceedings of ACM SIGCOMM'02*, Pittsburgh, PA.

Kotenko, I.V. (2005). Agent-Based Modeling and Simulation of Cyber-Warfare between Malefactors and Security Agents in Internet, *Proceedings of 19th European Simulation Multiconference "Simulation in wider Europe"*.

Kotenko, I.V. & Ulanov, A.V. (2005). Agent-based simulation of DDOS attacks and defence mechanisms. *Journal of Computing*, Vol.4, Issue 2.

Kotenko, I.V. & Ulanov, A.V. (2006). Agent Teams in Cyberspace: Security Guards in the Global Internet, *Proceedings of CYBERWORLDS'2006*.

Kotenko, I. (2007). Multi-agent Modelling and Simulation of Cyber-Attacks and Cyber-Defence for Homeland Security, *Proceedings of IDAACS'2007*. Dortmund, Germany.

Lefevre, V.A. (2003). *Reflexion*. Moscow, "Kognito-Center" (in Russian).

Mahadevan, P.; Krioukov, D.; Fomenkov, M.; Huffaker, B.; Dimitropoulos, X.; Claffy, K. & Vahdat, A. (2005). *Lessons from Three Views of the Internet Topology*. Technical Report, CAIDA.

Mirkovic, J.; Prier, G. & Reiher, P. (2002). Attacking DDoS at the Source, *Proceedings of ICNP*, Paris, France, 2002.

Mirkovic, J.; Dietrich, S.; Dittrich, D. & Reiher, P. (2004). *Internet Denial of Service: Attack and Defence Mechanisms*. Prentice Hall PTR.

Mirkovic, J.; Robinson, M.; Reiher, P. & Oikonomou, G. (2005). Distributed Defence Against DDOS Attacks, *Technical Report CIS-TR-2005-02*. University of Delaware. CIS Department.

Negoita, M.; Neagu, D. & Palade, V. (2005). *Computational Intelligence Engineering of Hybrid Systems*. Springer Verlag.

OMNeT++ (2010). http://www.omnetpp.org/

Papadopoulos, C.; Lindell, R.; Mehringer, I.; Hussain, A. & Govindan, R. (2003). Cossack: Coordinated suppression of simultaneous attacks, *Proceedings of DISCEX III*.

Paruchuri, P.; Bowring, E.; Nair, R.; etc. (2006). Mutiagent Teamwork: Hybrid Approaches. *Computer society of India Communications*.

Peng, T.; Christopher, L. & Kotagiri, R. (2003). Protection from Distributed Denial of Service Attack Using History-based IP Filtering, IEEE Conference on Communications.

Perumalla, K.S. & Sundaragopalan, S. (2004). High-Fidelity Modeling of Computer Network Worm, *Proceedings of 20th Annual Computer Security Applications Conference (ACSAC'04)*.

Silva, F.; Endler, M.; Kon, F.; Campbell, R.H. & Mickunas, M.D. (2000). Modeling Dynamic Adaptation of Distributed Systems, *Technical Report UIUCDCS-R-2000-2196*, University of Illinois at Urbana-Champaign.

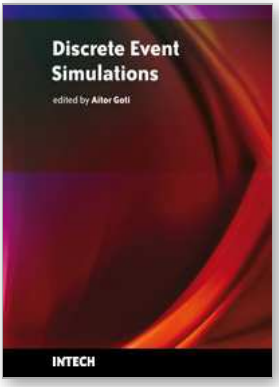
Tambe, M. (1997). Towards flexible teamwork. Journal of AI Research, Vol. 7.

Vilain, M. (1990). Getting Serious about Parsing Plans: A Grammatical Analysis of Plan Recognition, *Proceedings of the Eighth National Conference on Artificial Intelligence*, Cambridge, MA,

Want, R.; Pering, T. & Tennenhouse, D. (2003). Comparing autonomic and proactive computing. *IBM Systems Journal*, Vol.42, No.1.

Wellman, M.P. & Pynadath, D.V. (1997). *Plan Recognition under Uncertainty*, Unpublished web paper.

Zou, C.C.; Duffield, N.; Towsley, D. & Gong, W. (2006). Adaptive Defence against Various Network Attacks. *IEEE Journal on Selected Areas in Communications*: High-Speed Network Security (J-SAC), Vol. 24, No. 10.



Discrete Event Simulations
Edited by Aitor Goti

ISBN 978-953-307-115-2
Hard cover, 330 pages
Publisher Sciyo
Published online 18, August, 2010
Published in print edition August, 2010

Considered by many authors as a technique for modelling stochastic, dynamic and discretely evolving systems, this technique has gained widespread acceptance among the practitioners who want to represent and improve complex systems. Since DES is a technique applied in incredibly different areas, this book reflects many different points of view about DES, thus, all authors describe how it is understood and applied within their context of work, providing an extensive understanding of what DES is. It can be said that the name of the book itself reflects the plurality that these points of view represent. The book embraces a number of topics covering theory, methods and applications to a wide range of sectors and problem areas that have been categorised into five groups. As well as the previously explained variety of points of view concerning DES, there is one additional thing to remark about this book: its richness when talking about actual data or actual data based analysis. When most academic areas are lacking application cases, roughly the half part of the chapters included in this book deal with actual problems or at least are based on actual data. Thus, the editor firmly believes that this book will be interesting for both beginners and practitioners in the area of DES.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Igor Kotenko (2010). Agent-Based Modeling and Simulation of Network Infrastructure Cyber-Attacks and Cooperative Defense Mechanisms, Discrete Event Simulations, Aitor Goti (Ed.), ISBN: 978-953-307-115-2, InTech, Available from: http://www.intechopen.com/books/discrete-event-simulations/agent-based-modeling-and-simulation-of-network-infrastructure-cyber-attacks-and-cooperative-defense-

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821