Conference on Cyber Conflict
Proceedings 2010
C. Czosseck and K. Podins (Eds.)
CCD COE Publications, 2010, Tallinn, Estonia.

AGENT-BASED MODELING AND SIMULATION OF BOTNETS AND BOTNET DEFENSE

Igor Kotenko, Alexey Konovalov, and Andrey Shorov[1]

Laboratory of Computer Security Problems, St. Petersburg Institute for Informatics and Automation of Russian Academy of Sciences, St. Petersburg, Russia

Abstract: Nowadays we are witnesses of the rapid spread of botnets across the Internet and using them for different cyber attacks against our systems. Botnets join a huge number of compromised computers in the Internet and allow using these computers for performing vulnerability scans, distributing denial-of-service (DDoS) attacks and sending enormous amounts of spam emails. It is a very complex task to detect such botnets and protect against their attacks. The paper considers the approach to the investigation of botnets and botnet defense mechanisms. The approach is based on the agent-based simulation of cyber attacks and cyber defense mechanisms, which combines discrete-event simulation, multi-agent approach and packet-level simulation of network protocols. The various methods of botnet attacks and counteraction against botnet DDoS attacks are explored by representing botnets and botnet defense components as agent teams using the software simulation environment under development. Agents are supposed to collect information from various sources, use different knowledge, forecast the intentions and actions of other agents, try to deceive the agents of competing team, react to actions of other agents. The teams of defense agents are able to cooperate as the defense system components of different organizations and Internet service providers (ISPs). The paper outlines the common framework and implementation peculiarities of the simulation environment as well as the experiments aimed on the investigation of botnets and botnet DDoS defense mechanisms.

Keywords: cyber conflicts, cyber defense, botnets, Internet attacks and defense, DDoS, modeling and simulation, packet-based simulation, agent-based systems

1 39, 14th Liniya, SPIIRAS, St. Petersburg, 199178, Russia, Emails: {ivkote, konovalov, ashorov}@comsec.spb.ru.

INTRODUCTION

In April-May 2007, Estonia experienced several weeks of coordinated cyber attacks against its financial and sociopolitical institutions. As many authors declared, in this case Europe experienced its first information war (Blank, 2008). These attacks included huge denial of service attacks inspired by botnets.

A botnet is a computer network that consists of a certain number of hosts, where bots are run. A bot is standalone software. Most often the bot in the botnet is a program that is surreptitiously installed on the victim's computer and allows an attacker to perform some actions using the resources of the infected computer. Today, botnets confidently occupy the leading positions in the list of current threats to the Internet.

With the advent of botnets, malefactors have got access to millions infected computers of users, and the number cyber crimes has increased by hundreds of times. According to the FBI for October, 2009, losses because of botnets have reached about $100 million. Now experts observe the amplification of competition in the market of botnets, for example, at the end of 2009 – the beginning of 2010 a lot of new botnet programs, such as Filon, Clod, Buga, Spy Eye, has appeared (Truhanov, 2010).

The distinctive features of modern botnets as tools for cyber crime are a wide variety of possible targets of attacks, including the theft of personal or any other confidential data (theft of money from electronic invoices, credit card numbers, etc.), spam, forced advertise show, DDoS attacks, use of infected computers in their own purposes, compromise of legitimate users, cyber blackmail, fraud of rating tracking systems (for example, attack ClickFraud). The spectrum of the attacks, implemented by means of botnets, is rather extensive. Botnets are potentially suitable for attacks as in areas directly connected with cyber security, as well as in sphere of social engineering (Colarik, 2006).

Functioning of botnets is characterized by the simultaneous actions of a great number of software agents in the interests of malefactors. In most cases, a malefactor gains complete control over the resources of infected computers and can freely use them in almost any of their own interests. The prominent aspect of botnet use is the orientation of attackers usually on political goals or financial results, and, as a consequence, a high level of attack preparation. It stipulates considerable difficulty in identifying the organizers of botnets and in neutralizing them.

The last years, in particular, in Russia the following tendencies for botnets were observed (Lopuhin&Sachkov 2009):

- Enlargement – small botnets evolved in a larger ones; there was their associa-

tion and escalating of force for the opportunity of more powerful attacks;

- Decentralization – the command and control centers were transferred to the countries of "the third world" and decentralized;
- Occurrence of nonprofessional botnets by using special toolkits for their creation; the special knowledge for creation and management of such botnet is not required;
- Professional botnets began to use advanced technologies for command, control, communication, intelligence and maintenance of anonymity. In particular, some botnets began to use portknocking authentication technology (www.portknocking.org) etc.

All this emphasizes the urgency of research on protection against botnets. One of the major tasks of such research is an investigative modeling and simulation of botnets and defense mechanisms against them. The purpose of the research is the development of effective methods and means for botnet counteraction.

The paper considers *an approach to investigation of botnets and botnet defense mechanisms*. The approach is *based on the agent-based simulation of cyber attacks and cyber defense mechanisms, which combines discrete-event simulation, multi-agent approach and packet-level simulation of network protocols*. Initially this approach was suggested for network attack and defense simulation in (Kotenko&Ulanov 2006 a,b, 2007, 2008). In the present paper as against other works of authors the various methods of botnet attacks and counteraction against botnets are explored by representing attack and defense components as agent teams.

The global goal of our research is to *develop the usable common framework and simulation environment (integrated software tool) for analysis of botnets and botnet defense mechanisms*. The paper makes the greatest accent on specifying the scenarios of botnet functioning and defense mechanisms, describing the agent-based simulation environment under development and presenting the results of experiments carried out.

The paper is structured as follows. The first section describes the relevant papers and the features of proposed approach. Second section outlines the architecture and current implementation of agent-based simulation environment. Third section presents the configuration of simulation environment for experiments. Forth section considers the examples of experiments conducted. Conclusion outlines the main results and future work directions.

1. RELATED WORKS AND THE APPROACH TO SIMULATION

Current research on botnets and botnet defense can be considered mainly in two categories – botnet detection/response techniques and botnet measurement (Bailey, et al., 2009, Liu, et al., 2009, Strayer, et al., 2008). Botnet detection can be implemented, for example by detection via bot cooperative behaviors (Gu, et al., 2008a,b, 2007, Karasaridis, et al., 2007, Strayer, et al., 2006), detection by signatures of botnet communication process (Binkley&Singh 2006, Goebel&Holz 2007), and detection and response to attacks (Chen&Song 2005, Mirkovic, et al.,2004, 2005, Xie, et al.,2008). Measurement papers allow understanding the botnet phenomenon and the characteristics of specific types of botnets (Bailey, et al., 2009). Examples of papers on botnet measurement are (Dagon, et al.,2007, Gianvecchio, et al.,2008, Grizzard, et al., 2007, Kanich, et al., 2008, Rajab, et al.,2007, Wang, et al., 2007, Zhu, et al., 2008).

The most dangerous classes of attacks, which are the basic attack means of botnets, are *DDoS attacks* (Mirkovic, et al.,2004). *Traditional defense* from such attacks includes detection and reaction mechanisms. To detect abnormal network characteristics, many methods can be applied (for instance, statistical, cumulative sum, pattern matching, etc). The examples of detection methods are Hop counts Filtering (HCF), Source IP address monitoring (SIPM), Bit per Second (BPS), etc. As a rule, the reaction mechanisms include filtering, congestion control and traceback. As the detection of Botnet DDoS is most accurate, when it is close to the victim hosts, and the separation of legitimate is most successful close to the sources, adequate victim protection against Botnet DDoS to constrain attack traffic can only be achieved by *cooperation of different distributed components* (Mirkovic, et al., 2005). There are a lot of architectures for distributed cooperative defense mechanisms, e.g. Server Roaming, Market-based Service Quality Differentiation (MbSQD) (Mankins, et al., 2001), Transport-aware IP router architecture (tIP) (Wang&Shin 2003), Secure Overlay Services (SOS) (Keromytis, et al., 2003), ACC pushback, COSSACK (Papadopoulus, et al., 2003), Perimeter-based DDoS defense (Chen&Song 2005), DefCOM (Mirkovic, et al., 2005), Gateway-based (Xuan, et al., 2001).

The approach to botnets and botnet defense modeling and simulation, developed in the paper, is based on works in various fields. The basis of the proposed approach is agent-based modeling and simulation. Its essence is in representing the entities of subject area as particular autonomous intelligent agents. A set of intelligent agents, with simple functions, in process of their activities can be self-organized into a system with complex behavior needed for modeling and simulation of botnets and botnet defense.

The variety of frameworks and architectures for multi-agent modeling and simulation of distributed complex systems was developed, e.g. shared plans theory (Grosz&Kraus 1996), joint intentions theory (Cohen&Levesque 1991), hybrid approach (Tambe, 1997), there were implemented various software multi-agent environments (Macal&North 2005, Marietto, et al., 2002). The approaches based on belief-desire-intention (BDI), distributed constraint optimization (DCOP), distributed POMDPs, and auctions or game-theoretic (Tambe, 1997) are emphasized. Different mechanisms for collaborative agent team maintenance are suggested (Kaminka, et al., 2007). The main task of these approaches is *to provide the optimal interaction of heterogeneous components to reach some high level goal*. These methods, models and tools have been applied in different subject domains (Haque, et al., 2005, Jennings 1995, Kaminka&Frenkel 2005, Tambe, 1997).

The property of self-organization of agents into teams makes it appropriate to use them in problems related to cooperative games (Russell&Norvig 2009). Agents can get information about other agents, as well as on the state of environmental factors. Based on these data and on the basis of past experience and their functional role, the agent makes a decision regarding its future behavior. Thus, for effective decision-making, it is required to find an effective way to represent knowledge about the model of the external world inside the agent, using the theory of knowledge representation and the logic of reasoning. Goals may be common to a set of agents. Successful achievement of such objectives requires decision of problems related to cooperation and coordination of agents. Often the objectives for the group of agents are mutually exclusive; therefore, to build effective collective strategies, it is necessary to use the theory of antagonistic games.

The properties of the network environment, which is a space for network device operation, are characterized by high variability. The account of varying properties of the environment in the agent model stipulates the need to its adaptation and learning. Using agents as independent structural units does not exclude the presence of knowledge that is external to the agents and shared by them. Such knowledge can be a repository of collective and multi-level goals, the successful achievement of which requires the use of planning methods.

In the paper, the botnet life cycle and the process of botnet containment by defense mechanisms are simulated. Similarly to botnet structure, the structure of defense mechanisms is implemented by the subnet of defense components (agents). Both subnets are part of the overall computer network imitating the behavior of a certain segment of the Internet.

The paper also presents a model of the computer network topology in the form of a random graph, parameterized by statistical data obtained by measuring the topology of the Internet (Zhou, et al., 2006). Traffic is simulated on the level of individual

network packets, and is implemented by modeling the behavior of network applications. The behavior of network applications has a stochastic nature and is specified by a number of statistical parameters, whose values were derived from the study of network applications. The processes of sending, receiving and passing the network packets on communications channels are imitated through a discrete event simulation system.

The model of botnet, presented in the paper, is based on the results of earlier works conducted by several investigators (Bailey , et al., 2009, Barford&Yegneswaran 2007, Bradley&Harley 2007, Christodeorescu&Rubin 2007, Dungam&Meluick 2008, Mirkovic, et al., 2004, Strayer, et al., 2008). In particular, in some papers the structure of the botnet life cycle, the phases of botnet functioning as well as the technological and organizational aspects of each phase were outlined (Barford&Yegneswaran 2007, Anon. 2007, Mirkovic, et al., 2004).

In this paper, the models of botnets and botnet defense are specified in the form of a common model of counteraction between the two classes of teams – the attack team and the defense team. Each team represents a subset of computer network nodes, identified as agents, and having a common collective goal. Attack team includes agents belonging to the botnet and implementing actions aimed at achieving the collective goal – providing the vital activity of botnets. An example of the collective goal of the attack team is a DDoS attack against some pre-identified resource. Similarly, the defense team is made up of agents, performing the defense functions and having a collective goal to oppose the botnet. It is supposed that the team of botnet agents evolves by generating new instances and types of attacks and attack scenarios to overcome the defense. The team of defense agents adapts to the botnet actions by changing the security policy and forming new instances of defense methods and profiles.

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 to represent the investigated processes. They include particular team ontologies, agent basic functions, agent classes, agent protocols and behavior 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 instruc-

tion 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. Agent team establishing protocol is the part of procedures for monitoring and recovery of agent functionality. *Scenarios* represent various stages of team actions. Adaptation procedures are implemented in scenarios to act depending on other team actions and environment reaction. Agent team behavior scenarios ensure action consistency maintenance. (Ulanov & Kotenko 2008)

Models of team interactions include the models of antagonistic competing and team cooperation. *Model of antagonistic competing* lies in the basis of competing teams' interaction. This model defines the goals, sub-goals, 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 defense simulation it is possible to exchange attack signatures, network traffic data, filtering requests, etc.

2. SIMULATION ENVIRONMENT

To implement the proposed approach, a software environment is required, that has a wide range of opportunities to support network simulation. In the first place, we need a possibility to simulate the network systems with arbitrary topology and communication processes between different nodes at the level of discrete events. To simulate real-world communication scenarios, observed in the Internet, we must have the models of protocols and network applications.

The description of individual and group behavior of agents as well as experiment parameters supposes the presence of a high level language to specify such behavior scenarios and parameters.

The authors of the paper are trying to use and develop a multi-level instrumental environment for simulation of network processes. This environment is a software package that includes a discrete event simulator, implemented by low-level language, and a number of components that realize the components of higher levels.

The lower level provides a possibility to simulate the chronologically ordered sequences of events, propagating in network structures. Intermediate levels on the basis of the lower level implement the components related to the specifics of the Internet, including the models of protocols and standard network applications. Inter-

mediate levels are the basis for constructing components of a higher level, such as, for example, the level of intelligent agents. All modules and components of the simulation environment are in conjunction with the I/O subsystem and, thus, through this subsystem can communicate with external data sources and with the system operator. Each level is implemented as a separate function library with a documented interface. Such interface ensures the opportunity to interact with this library from the side of other components.

The architecture of simulation environment consists from four main components (Figure 1).

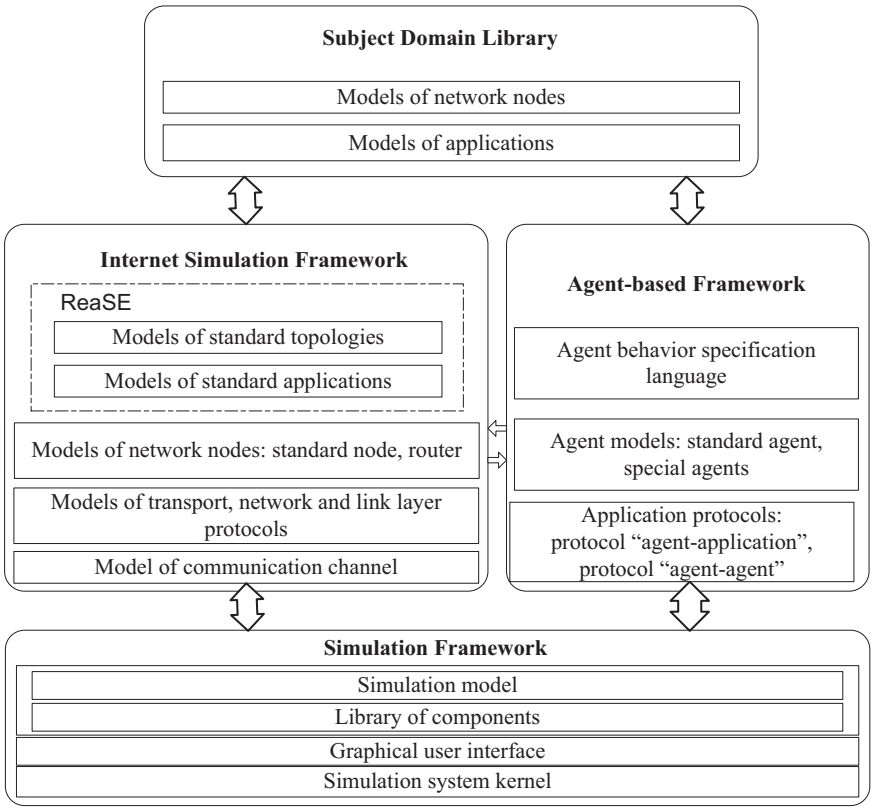

Figure 1. Architecture of simulation environment

Simulation Framework is a discrete event simulation system. It provides tools for modeling chronologically ordered sequences of discrete events. Simulation Framework implements the basic models of random event distributions and the basic mod-

els of queues with priorities and the collection of statistics. The possibilities of basic model data input/output and basic features for processing the results of experiments are provided.

Internet Simulation Framework is a set of modules which allow simulating nodes and protocols of the Internet. It contains the modules that form realistic network topologies, the models of network applications with behavior close to the behavior of real network applications, as well as the models of transport, network and link layer protocols. Thus, this component provides the models of computer network as a network with nodes that include a stack of TCP/IP-protocols. Each protocol is implemented as an independent module. Internet Simulation Framework also contains modules for automatic construction of standard networks based on the set of defined parameters and their automatic configuration. In current version this component uses the library ReaSE (Gamer&Scharf 2008).

Representation of network elements as intelligent agents is realized by using *Agent-based Framework*. This component is a library of modules that specify intelligent agents and common scripts of their behavior, implemented in the form of models of services and applications embedded in the models of network nodes. The component also contains the models of application layer protocols, which provide communication between agents and interaction of agents with application models. In addition, the component includes a high-level language interpreter to manage agents and a transmitting module, which converts the commands of the language into the sequence of intelligent agent actions.

Subject Domain Library is a library, which serves to simulate the processes of the subject area. It includes modules that complement the functionality of IP node, including filter tables, packet analyzers, models of legitimate users, etc.

Using several different components, such as the simulator OMNeT++ (www.omnetpp.org), the libraries INET Framework and ReaSE, and our own software components (Kotenko&Ulanov 2006a,b, 2007), the proposed architecture has been implemented for agent-based simulation of botnets and defense mechanisms against them. Models of agents, implemented in Agent-based Framework, include an agent "legitimate client", an agent "legitimate server", attack agents and defense agents. Subject Domain Library contains various models of nodes, for example, attacker, firewall, etc., as well as application models (mechanisms for implementation of attacks and protection, packet sniffers, filter tables).

OMNeT++ is a discrete event simulation system. In the proposed architecture, it is a lower level component. OMNeT++ provides message exchange between the components simulated, experiment visualization, interaction with the user, and debugging the states of objects and sequences of model events. In addition, OMNeT++ provides

a special language to describe connections between components simulated, thus describing the network topology, and allows specifying the experiment parameters. The structure of individual participants, involved in message exchange, is described as C++ classes; the logic of message processing by specific actors is determined by algorithms implemented in C++ class methods. Using the high-level language to specify the static relations between components and the environment parameters allows achieving high flexibility in configuring simulated components, because it does not require translating configuration scripts into binary code. On the other hand, the usage of C++ language for message processing is characterized by low overhead, because the handler code is compiled into a high-performance machine code. Thus, an approach, based on combining the low and high level languages, provides a high performance with a sufficiently flexible configuration mechanism that may be important when conducting multiple experiments.

The main window of OMNET++ graphical development environment (for version 4.0) is shown in Figure 2.

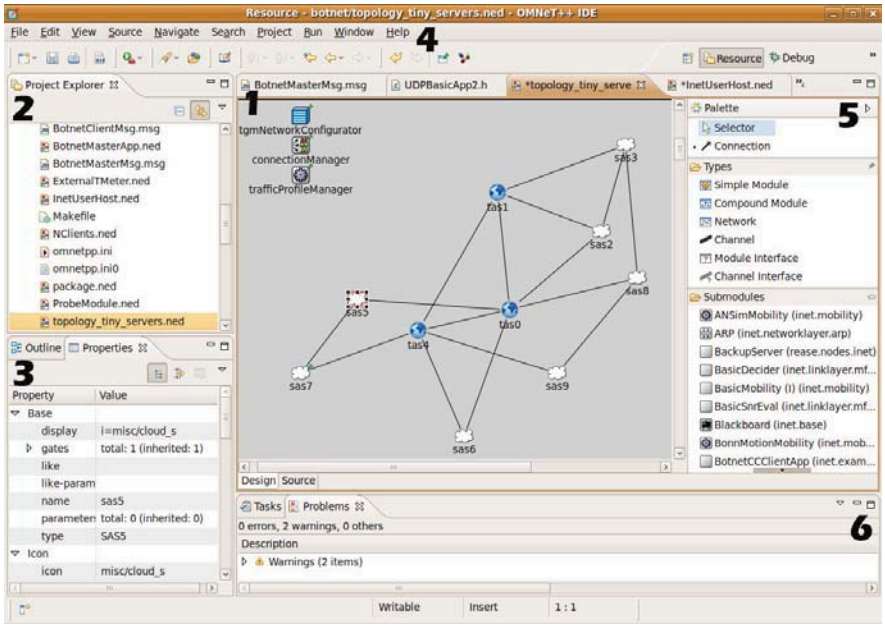

Figure 2. Main window of OMNET++ 4.0 development environment

The development environment window is divided into several zones, which contain a variety of visual tools (sub windows). The main simulation window is marked with 1. It contains the structure of the current component or the view of the whole net-

work. In the upper left corner (2) there is a list of files included in the current model. Below (3), there is a table of properties of currently active object (in the figure it is an object with the name sas5), which is highlighted with a frame on the main window.

Also there is a palette of components (5), available for insertion into the current model, and a console with various information to assemble the model (6). The main menu and toolbar are situated at the top of the development environment. Objects, involved in simulation, are represented by appropriate images with the specified object names. Connections between objects are shown by solid lines.

Figure 2 shows the objects that have connections with other objects and the objects without connections. Unconnected objects are not involved in message exchange and, as a rule, perform various service functions, such as, for example, the configuration of other objects or collecting the statistics.

An example of the model representation in experiments is shown in Figure 3.

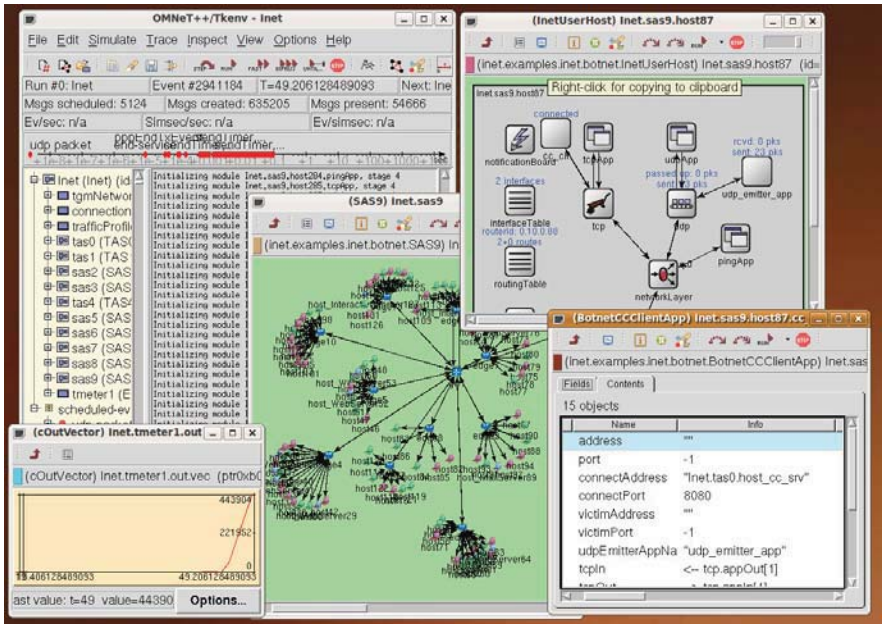

Figure 3. Representation of the model in experiments

In the upper left corner you can see the main panel that displays the components included in the model and control elements that allow the user to interact with them. In addition, the main panel consists of the model time control elements, which allow executing the model step by step or in maximum fast mode. There are also the con-

trol elements to perform an efficient search for the entity of interest and subsequent editing of its condition.

Figure 3 shows a fragment of the simulated network, where the router models are shown as cylinders with arrows, and the host models are in the form of computers having different colors.

Color displays the status of nodes in the botnet. Green represents the nodes that are part of botnet and have a connection to the command center, red – the nodes which have received the command to attack the target. Nodes that are not included in the botnet do not have a color tint.

As an example, Figure 3 also shows the window of a host representation (top right), the window for editing the parameters of the object "bot-client" (bottom right), and the window of current experimental results (bottom left), showing in graphical form the value of one of the investigated parameters.

3. IMPLEMENTATION CONFIGURATION

3.1 NETWORK TOPOLOGY

The network topology is simulated on *two levels of detail*.

On the first level, the network topology on the autonomous system (AS) level is simulated. A number of authors recommend using the PFP (Positive Feedback Preference) method (Zhou, et al., 2006) to simulate the Internet on the autonomous system level. This method allows the most plausible representation of statistical properties of the Internet topology segments. In the paper we describe the experiments in which a network consisting of 5-10 autonomous systems (AS-level topology) is simulated. We generate a graph of autonomous system level with the following parameters: Transit Node Threshold = 10, P = 0.4, Delta = 0.04 (Zhou, et al., 2006). Connections of transit AS are implemented through the communication channel with the bandwidth dr = 10000 Mbit/seconds and the delay d=1 microseconds. Connections of other AS are implemented with dr = 5000 Mbit/seconds and d = 1 microseconds.

On the second level of simulation, for each autonomous system the internal topology (Router-level topology) is simulated. In the paper we use so called HOT (Heuristically Optimal Topology) model (Li, et al., 2004) with the following parameters: Min nodes = 20, Max nodes = 25, Core ratio = 0.05, Core cross link ratio = 0.2, Min hosts per edge = 10, Max hosts per edge = 20 (Li, et al., 2004). Each autonomous system includes approximately 300 end-nodes (Figure 4). The equipment of nodes has the

types “router” or “host’. The equipment “router” has only one functional role – “router”. The equipment “host” is represented by the following set of functional roles: web server, web client, mail server, server of multimedia content, “command center” server, “vulnerable service” (potential “zombie” machine), “master”, IP-filter. At each node the model of standard protocol stack is installed. It includes the protocols PPP, LCP, IP, TCP, ICMP, ARP, UDP. Also, depending on the functional role, the models of network applications are installed. They implement application-level protocols. For each protocol, the appropriate adjustment of its parameters is fulfilled. For example, for IP protocol the corresponding adjustment of routing tables is carried out in accordance with the principle of minimizing the number of intermediate nodes on the route of IP-packet.

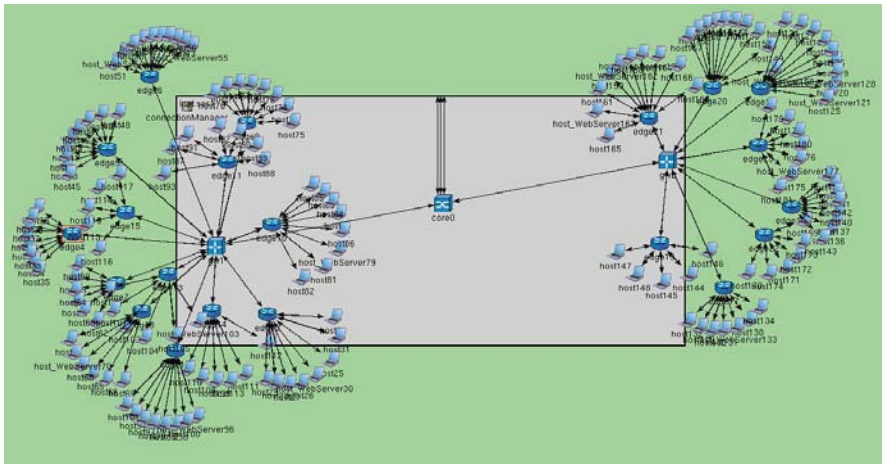

Figure 4. Example of an autonomous system representation

3.2 ATTACK TEAM CONFIGURATION

Attack teams include the following types of nodes: “master”, “command center”, “target”, “zombies”. Node “master”, by sending different commands, sets the goals of the botnet and controls the behavior of the network at the highest level. Node “command center” carries out the delivery of commands received from the “master” to nodes “zombies”. Nodes “zombie”, receiving the commands from the “command center”, immediately carry out actions under the orders of the “master”. In experiments we use a single node “master”, one or several nodes “command center” and a set of nodes with vulnerable software, which can potentially turn into “zombie” machines.

3.3 CONFIGURATION OF LEGITIMATE HOSTS

To generate a legitimate traffic, the set of nodes "server" (in the amount of 10% of the total number of nodes) is determined. The nodes "servers", in response to a request from the nodes "clients", generate the traffic statistically similar to traffic of standard web server (Gamer&Scharf 2008). Vulnerable hosts are determined randomly. They represent about 40% of the total number of nodes. One of examples of a vulnerable service is based on UDP protocol and uses port 80.

3.4 DEFENSE TEAM CONFIGURATION

Defense teams are represented by the following common classes of agents: information processing ("sampler"); attack detection ("detector"); filtering ("filter"); investigation ("investigator"); rate limitation ("limiter"). Samplers collect and process network data for anomaly and misuse detection. Detector coordinates the team, correlates data from samplers, and detects 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 DDoS defense. 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. (Kotenko & Ulanov 2008)

In experiments, the method of Source IP address monitoring (SIPM) is used as the defense mechanism. It is based on the assumption that during DDoS attacks in the passing traffic, the number of new addresses used for connection with the attacked resource grows quickly.

The module, which implements the defense mechanism, may be in one of two modes: training or working (i.e. anomaly detection and traffic filtering).

In the training mode, the module intercepts the traffic and determines the number of different IP-addresses involved in the communication for a certain time period (parameter tshift). In the experiments the value of tshift is 2 seconds. The data obtained in the training mode are taken as typical traffic values in the node and then are used in the process of anomaly detection.

In the working mode, the module calculates the same parameters and compares them with typical values. When a significant excess of observed nominal values is observed, the protection module generates an anomaly detection signal and provides selective filtering of packets with new IP addresses.

3.5 REALIZATION OF SCENARIOS

To perform the experiments we have realized several scenarios of botnet functioning (including scenarios of botnet propagation, botnet management and attack realization), botnet containment and attack counteraction, and network legitimate activity.

Scenarios of botnet propagation involve scenarios of looking for new nodes suitable for compromise, their identification, subsequent compromise, and connecting the infected nodes to the botnet.

Scenario of botnet propagation used in the experiments is based on the model of a network worm spreading through the exploitation of the vulnerability of network services. After activation of a vulnerable service, the computer is considered infected. An example of a scenario is illustrated in Figure 5. After infection, the infected computer icon turns yellow.

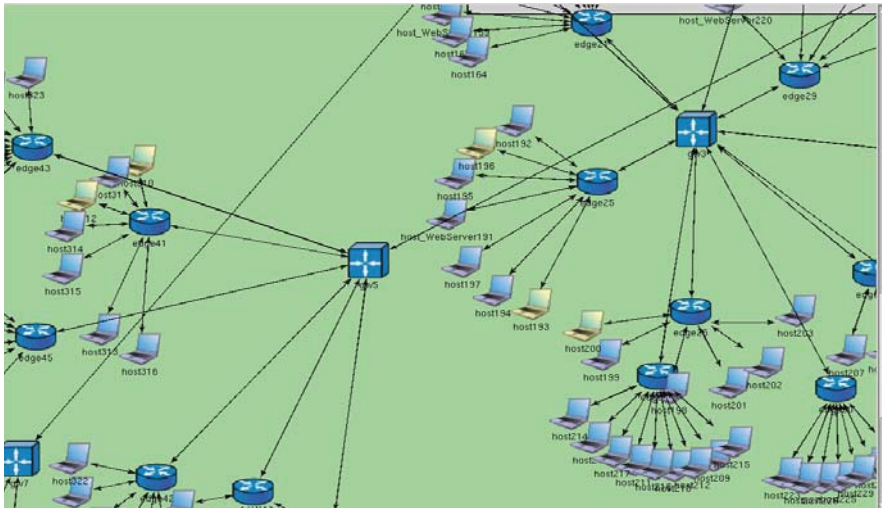

Figure 5. Infected computers

Scenario of connecting the infected nodes to botnet is specified by the procedure of sending the message about the new node status to the server "command center" and then pending the receipt of commands from the server.

One of the examples of implemented *scenarios of botnet attack realization* is an attack "UDP Flood", directed to some node (subnet), the IP-address of which is specified in the attack start command.

We implemented several *scenarios of botnet containment and attack counteraction*,

directed on protection against DDoS attacks: without cooperation; DefCOM-based; COSSACK-based and full cooperation.

In defense scenario *without cooperation* only one defense agent team is used. This team is represented by the following common classes of agents: "sampler", "detector", "filter", "investigator", and "limiter".

In other defense scenarios we use several cooperating defense agent teams which protect different segments of the computer network.

The following agent classes are proposed to introduce in compliance with *DefCOM* architecture (Mirkovic, et al., 2005): "Alert generator" agent is based on a "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 a "limiter" agent. It can drop the packets destined to the attack target providing some volume of traffic. Agent "Classifier" is based on a "filter" agent that receives filtering data from the detector. This agent is able to filter the disclosed attack packets. It also marks the legitimate packets to let "limiter" pass them. 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 (Papadopoulus, et al., 2003) 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 an 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 an agent "detector". It makes the decision about attack due to data from "snort". Agent "filter" is used to simulate the filter on the router. It is deployed on the router and performs traffic filtering using data from "watchdog". "Watchdog"-level cooperation is used to transmit the filtering rules. 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 defense agents: "samplers", "detectors", "filters", and "investigators". Under full cooperation the team, which network is the attack victim, can receive traffic data from the samplers of

other defense teams and apply the filtering rules on the filters of other teams.

The algorithms of *network legitimate activity scenario* are based on generation of the model traffic with statistical parameters, similar to parameters of a real network traffic (Vishwanath&Vahadat 2006). They are executed by sub-scenario of session creation, using the parameters which depend on the type of generated traffic.

4. EXPERIMENTS

The investigation of attack and defense scenarios 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 the defense team, which network is the attack victim.

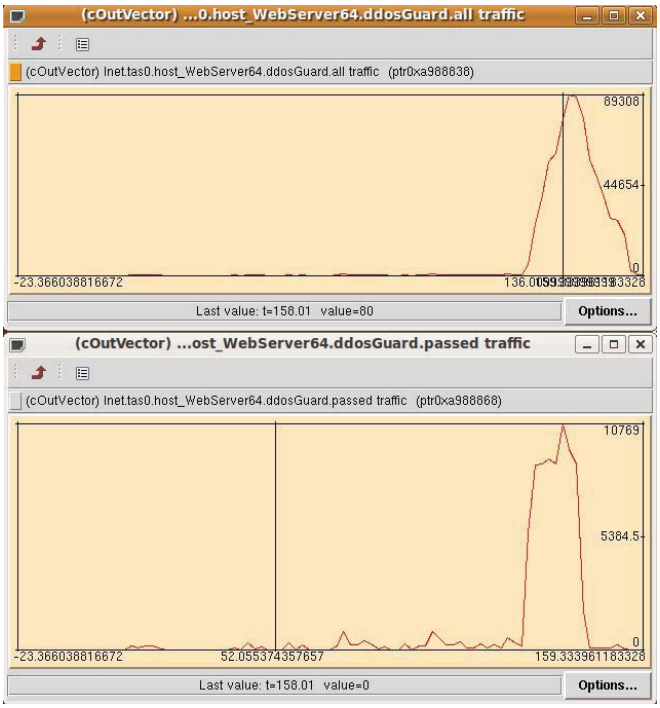

Figure 6. Examples of traffic levels before and after filtering

Under scenarios of botnet containment and attack counteraction, the defense system tries to separate the malicious traffic from the legitimate traffic, and, if possible, to filter out the malicious traffic. The results of the filtering process are estimated by

false positive (FP) and false negative (FN) rates. Examples of traffic levels before and after filtering for one of the experiments for *defense scenario without cooperation* are depicted in Figure 6. The figure shows that the traffic volume after filtering was reduced to nine times. The values of errors of first and second kind are as follows: FN = 0,09, FP = 0,002.

Figure 7 shows the attack traffic inside the attacked subnet for *scenarios of botnet containment and attack counteraction using COSSACK (triangles), DefCOM (dots) and full cooperation scheme (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).

Attack traffic for COSSACK is measured on the entrance to the defended subnet on the filter. The significant traffic increase is noticed in the beginning of attack. But in the area of 350 seconds the defense 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 defense 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 Figure 7. 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 defense 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.

The attack traffic inside the attacked subnet for the full cooperation scheme is represented with the crosses in Figure 7. Traffic is measured on the entrance to the defended subnet on the filter. The significant traffic increase is noticed in the beginning of attack. But in the area of 350 seconds the main defense 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). The system succeeds in decreasing the traffic much more due to permanent attack signatures renewing (400–450 seconds).

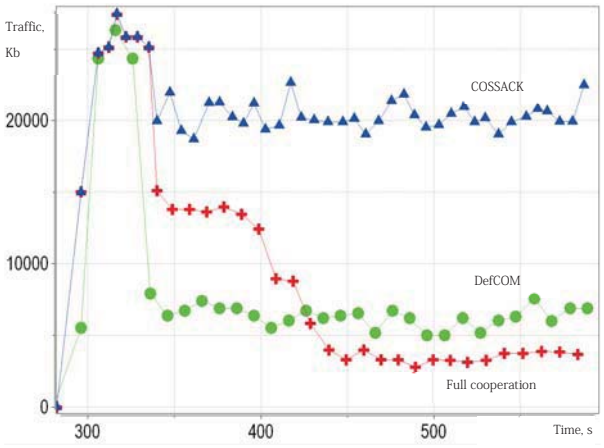

Figure 7. Attack traffic inside the attacked subnet for COSSACK, DefCOM and full cooperation

The experiments implemented demonstrated that full cooperation shows the best results on blocking the attack traffic. It uses several defense teams with cooperation on the level of filters and samplers. Samplers cooperation played the crucial role in defense.

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 defense network. It uses attack signatures transmission between agents to apply the filtering rules near the source. The communication overhead for cooperative defense 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.

Different *adaptation schemes of agent teams* were also studied. *Adaptation schemes* operate in the following way. Depending on attack state the defense team adapts the parameters of methods and cooperation reducing the defense cost. The simplest and not the most resource-intensive method is BPS. The defense team starts the defense-implementing BPS. When attack is detected the team continues to use the same method, if it allows the attack to be neutralized. If it fails, then the defense team applies the more complicated SIPM method. If it succeeds to stop the attack, then the defense team returns to BPS. If not – it will additionally use HCF. Conducted experiments showed that one can reach the best attack traffic blocking due to de-

fense teams cooperation.

Since sampler cooperation was the determinative in defense, it can be used without applying full cooperation during which high teams interaction traffic is observed.

Attack team redistributes the attack intensity between daemons and changes the address spoofing technique to minimize the amount of attack packets and reduce the probability of attack agents' exposure by defense agents. At first the team having many daemons distributes the load equal between them and does not use address spoofing (not to draw suspicion upon themselves from firewall in their subnet). If after the defense team actions some of the daemons will be defeated, the attack team will raise the load to the remaining daemons (to save the given attack intensity) and apply the address spoofing technique to avoid detection. If the remaining daemons are not defeated the team will continue the attack in the former mode.

5. CONCLUSION

This paper proposed the approach to simulation of botnets and defense against botnets in the Internet. Botnets and botnet defense is examined by interaction of different agents teams that can be in the relation of antagonistic and non-antagonistic competing or (and) various kinds of cooperation. The main results of the paper consist in specification of formal framework for botnet analysis and implementing the software simulation tool for packet-level agent based simulation of botnets attack and defense. Environment for the agent-oriented simulation was developed on the basis of OMNeT++ INET Framework.

This software simulation environment has been used for investigation of various cooperative distributed defense schemes against botnet DDoS attacks. The conducted experiments showed the availability of the proposed approach for simulation of complex botnets and defense against botnets and security analysis of projected networks. The experiments also showed that the use of cooperation of several defense teams leads to the essential raise of defense effectiveness.

The approach used in the paper allows simulating and investigating various kinds of botnets and botnet defense mechanisms. *We suppose that in the context of cyber conflicts the approach and simulation tool under development can be used for analyzing current and future defense mechanisms as well as be applied for "laboratorial" forensic investigation of botnets and network attacks fulfilled.*

Future work is related to comprehensive formal specification of botnets and defense mechanisms, deep analysis of cooperation effectiveness of various attack and defense teams and inter-team interaction, the implementation of adaptation and

self-learning defense to protect against manipulation by attackers, the expansion of attack and defense library to investigate more complicated scenarios of counteraction between botnets and botnet defense mechanisms, and the investigation of new defense mechanisms.

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 testbed combining a hierarchy of macro and micro level models of botnets and botnet defense (analytical, packet-based, emulation-based), and real small-sized networks.

The important part of future research is providing also numerous experiments to study various botnet attacks and the effectiveness of prospective defense mechanisms against botnet formation, propagation, attack detection, and response including tracing the source of attacks and botnet destruction.

ACKNOWLEDGMENTS

This research is being supported by the grant of Russian Foundation of Basic Research (# 10-01-00826-a), and the program of fundamental research of the Department for Informational Technologies and Computation Systems of the Russian Academy of Sciences (# 3.2).

REFERENCES

- Bailey, M., Cooke, E., Jahanian, F., Xu, Y., and Karir, M., 2009. A Survey of Botnet Technology and Defenses. In *Cybersecurity Applications & Technology Conference for Homeland Security*.
- Barford, P., Yegneswaran, V., 2007. An Inside Look at Botnets. In *Advances in Information Security*, Vol.27. Malware Detection. Springer.
- Binkley, J. R., Singh, S., 2006. An algorithm for anomaly-based botnet detection. In *2nd conference on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'06)*, San Jose, CA, July 2006.
- Blank, S., 2008. Web War I: Is Europe's First Information War a New Kind of War?. *Comparative Strategy*, Vol.27, Issue 3.
- Bradley, T., Harley, D., 2007. *Botnets: The Killer Web App*. Syngress Publishing, Inc. 2007.
- Chen, S., Song, Q., 2005. Perimeter-Based Defense against High Bandwidth DDoS Attacks. *IEEE Transactions on Parallel and Distributed System*, Vol.16, No.7.
- Christodorescu M., Rubin S., 2007. Can Cooperative Intrusion Detectors Challenge the Base-Rate Fallacy. *Advances in Information Security*, Vol.27. Malware Detection. Springer.
- Cohen P., Levesque, H.J., 1991. Teamwork. *Nous*, No.35.
- Colarik, A., 2006. *Cyber Terrorism: Political and Economic Implications*. Idea Group Inc.
- Dagon D., Gu G., Lee, C. P., and Lee, W., 2007. A taxonomy of botnet structures. In *Twenty-Third Annual Computer Security Applications Conference (ACSAC'07)*, Florida, USA, November 2007.
- Dunham K., Meluick, J., 2008. *Malicious Bots - An Inside Look into the Cyber-Criminal Underground of the Internet*. CRC Press.
- Gamer, T., Scharf, M., 2008. Realistic Simulation Environments for IP-based Networks. in *1st International Workshop on OMNeT++*. Marseille, France. 2008.
- Gianvecchio, S., Xie, M., Wu, Z., Wang, H., 2008. Measurement and classification of huamans and bots in internet chat. In *17th USENIX Security Symposium (Security'08)*, San Jose, CA, July 2008.
- Goebel, J., Holz, T., 2007. Rishi: Identify bot contaminated hosts by IRC nickname evaluation. In *First Workshop on Hot Topics in Understanding Botnets (HotBots'07)*, Cambridge, MA, April 2007.
- Grizzard, J. B., Sharma, V., Nunnery, C., Kang, B. B., Dagon, D., 2007. Peer-to-Peer Botnets: Overview and case study. In *First Workshop on Hot Topics in Understanding Botnets (HotBots'07)*, Cambridge, MA, April 2007.
- Grosz, B., Kraus, S., 1996. Collaborative Plans for Complex Group Actions. *Artificial Intelligence*, Vol.86, No.2.
- Gu, G., Perdisci, R., Zhang, J., Lee, W., 2008a. BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *17th USENIX Security Symposium (Security'08)*, San Jose, CA, July 2008.
- Gu, G., Porras, P., Yegneswaran, V., Frog, M., Lee, W., 2007. BotHunter: Detecting malware infection through ids-driven dialog correlation. In *16th USENIX Security Symposium (Security'07)*, Boston, MA, August 2007.
- Gu, G., Zhang, J., Lee, W., 2008b. BotSniffer: Detecting botnet command and control channels in network traffic. In *15th Annual Network & Distributed System Security Symposium (NDSS'08)*, San Diego, CA, February 2008.
- Haque, N., Jennings, N.R., Moreau, L., 2005. Resource allocation in communication networks using market-based agents. *International Journal of Knowledge Based Systems*, Vol.18, No.4-5.
- Jennings, N.R., 1995. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence*, Vol.75, No.2.
- Kaminka, G.A., Frenkel, I., 2005. Flexible teamwork in behavior-based robots. In *AAAI-05*.
- Kaminka, G.A., Yakir, A., Erusalimchik, D., Cohen, N., 2007. Towards Collaborative Task and Team Maintenance. In *AAMAS-07*.

- Kanich, C., Lechenko, K., Enright, B., Voelker, G. M., Savage, S., 2008. The Heisenbot Uncertainty Problem: Challenges in separating bots from chaff. In *First Usenix Workshop on Large-scale Exploits and Emergent Threats (LEET'08)*, San Francisco, CA, April 2008.
- Karasaridis, A., Rexroad, B., Hoeflin, D., 2007. Wide-scale botnet detection and characterization. In *First Workshop on Hot Topics in Understanding Botnets (HotBots'07)*, Cambridge, MA, April 2007.
- Keromytis, A.D., Misra, V., Rubenstein, D., 2003. SOS: An architecture for mitigating DDoS attacks. *Journal on Selected Areas in Communications*, No.21.
- Kotenko, I., Ulanov, A., 2006a. Agent Teams in Cyberspace: Security Guards in the Global Internet. In *International Conference on CYBERWORLDS (CW2006)*. IEEE Computer Society.
- Kotenko, I., Ulanov, A., 2006b. Agent-Based Modeling and Simulation of Network Softbots' Competition. In *Seventh Joint Conference on Knowledge-Based Software Engineering*, Amsterdam: IOS Press, Vol.140.
- Kotenko, I., Ulanov, A., 2007. Multi-agent Framework for Simulation of Adaptive Cooperative Defense against Internet Attacks. In *International Workshop on Autonomous Intelligent Systems: Agents and Data Mining (AIS-ADM-07)*. Springer, Vol.4476.
- Kotenko, I., Ulanov, A., 2008. "Packet Level Simulation of Cooperative Distributed Defense against Internet Attacks. In *16th Euromicro Conference on Parallel Distributed and Network-Based Processing (PDP 2008)*,
- Li, L., Alderson. D., Willinger, W., Doyle, J., 2004. A first-principles approach to understanding the internet's router-level topology. *ACM SIGCOMM Computer Communication Review*.
- Liu, J., Xiao, Y., Ghaboosi, K., Deng, H., and Zhang, J., 2009. Botnet: Classification, Attacks, Detection, Tracing, and Preventive Measures. *EURASIP Journal on Wireless Communications and Networking*, Vol.2009.
- Lopuhin, I. , Sachkov, I., 2009. *The brief analytical questionnaire on botnets in the Russian Federation for 2009 year*. March, 2009. Available at: http://www.securitylab.ru/analytics/370022.php [Accessed Feb. 15, 2010]. (in Russian)
- Macal, C.M., North, M.J., 2005. Tutorial on Agent-based Modeling and Simulation. In *2005 Winter Simulation Conference*.
- Mankins, D., Krishnan, R., Boyd, C., Zao, J., Frentz, M., 2001. Mitigating Distributed Denial of Service Attacks with Dynamic Resource Pricing. In *17th Annual Computer Security Applications Conference*.
- Marietto, M., David, N., Sichman, J.S., Coelho, H., 2002. Requirements Analysis of Agent-Based Simulation Platforms: State of the Art and New Prospects. *Lecture Notes in Artificial Intelligence*, Springer, Vol.2581.
- Mirkovic, J., Dietrich, S., Dittrich, D., Reiher, P., 2004. *Internet Denial of Service: Attack and Defense Mechanisms*. Prentice Hall PTR.
- Mirkovic, J., Robinson, M., Reiher, P., Oikonomou, G., 2005. Distributed Defense Against DDOS Attacks. *University of Delaware CIS Department Technical Report CIS-TR-2005-02*.
- Papadopoulos, C., Lindell, R., Mehringer, I., Hussain, A., Govindan, R., 2003. Cossack: Coordinated suppression of simultaneous attacks. In *DISCEX III*.
- Rajab, M. A., Zarfoss, J., Monrose, F., Terzis, A., 2007. My Botnet is Bigger than Yours (Maybe, Better than Yours): why size estimates remain challenging. In *First Workshop on Hot Topics in Understanding Botnets (HotBots'07)*, Cambridge, MA, April 2007.
- Russell, S., Norvig, P., 2009. *Artificial Intelligence: A Modern Approach* (3rd Edition). Prentice Hall.
- Strayer, W.T., Lapsely, D., Walsh, R., Livadas, C., 2008. Botnet Detection Based on Network Behavior. *Advances in Information Security,* Vol.36. Botnet Detection.
- Strayey, W. T., Walsh, R., Livadas, C., Lapsley, D., 2006. Detecting botnets with tight command and control. In *31st IEEE Conference on Local Computer Networks (LCN06)*, Tampa, Florida, November 2006.
- Tambe, M., 1997. Towards flexible teamwork. *Journal of AI Research*, No.7.
- Tambe, M., Bowring, E., Jung, H., et al., 2005. Conflicts in teamwork: Hybrids to the rescue. In *AAMAS-05*.

- Truhanov, A., 2010. Russian botnet wants to kill the competitor. 2010. Available at: http://safe.cnews.ru/news/top/index.shtml?2010/02/10/379202 [Accessed Feb. 15, 2010]. (in Russian)
- Ulanov, A., Kotenko, I., 2008. Simulation of Adaptable Agent Teams on the Internet. In *Proceedings of the 1st International Workshop on Logics for agents and mobility*
- Vishwanath, K.V., Vahdat, A., 2006. Realistic and responsive network traffic generation. In *Conference on Applications, technologies, architectures, and protocols for computer communications*.
- Wang, H., Shin, K.G., 2003. Transport-aware IP Routers: A Built-in Protection Mechanism to Counter DDoS Attacks. *IEEE Transactions on Parallel and Distributed Systems*, Vol.14, No.9.
- Wang, P., Sparks, S., Zou, C. C., 2007. An advanced hybrid peer-to-peer botnet. In *First Workshop on Hot Topics in Understanding Botnets (HotBots'07)*, Cambridge, MA, April 2007.
- Xie, Y., Yu, F., Achan, K., Panigrahy, R., Hulten, G., Osipkov, I., 2008. Spamming Botnets: Signatures and characteristics. In *ACM SIGCOMM'08*, Seattle,WA, August 2008.
- Xuan, D., Bettati, R., Zhao, W., 2001. A Gateway-Based Defense System for Distributed DoS Attacks in High Speed Networks. In *2nd IEEE SMC Information Assurance Workshop*, West Point, NY, 2001.
- Zhou, S., Zhang, G., Zhang, G., Zhuge, Z., 2006. Towards a Precise and Complete Internet Topology Generator. In *International Conference Communications, Circuits and Systems*.
- Zhu, Z., Lu, G., Chen, Y., Fu, Z. J., Roberts, P., Han, K., 2008. Botnet research survey. In *32nd Annual IEEE International Computer Software and Applications Conference*, Turku, Finland, July 2008.