

AGENT-BASED SIMULATION ENVIRONMENT AND EXPERIMENTS FOR INVESTIGATION OF INTERNET ATTACKS AND DEFENSE

Igor Kotenko and Alexander Ulanov
Computer Security Research Group
St. Petersburg Institute for Informatics and Automation
39, 14th Liniya, St. Petersburg, 199178, Russia
E-mail: {ivkote, ulanov}@iias.spb.ru

KEYWORDS

Agent-based Simulation, Network Simulation, Simulation of Internet attacks, Simulation Tools for Communications and Networks, Security, DDoS.

ABSTRACT

The paper considers the simulation environment which has been developed for comprehensive investigation of Internet attacks and defense (on an example of Distributed Denial of Service (DDoS) actions and protection mechanisms). This environment offers the following features: agent-oriented approach to simulation; packet-based simulation of attacks and defense systems, and capability to add new attacks and defense methods and investigate them. The suggested approach for simulation is considered. The main components of the simulation environment are specified. The testing methodology for defense investigation is described, and the results of experiments are outlined.

1. INTRODUCTION

One of the most dangerous classes of the Internet attacks is DDoS. Distributed, dynamical and cooperative character of such attacks complicates attack detection and protection. Realizing effective DDoS defense system is a very complicated problem. Effective defense includes the mechanisms of attack prevention, attack detection, tracing the attack source and attack counteraction (Mirkovic et al. 2004; Xiang et al. 2004; Yuan et al. 2005). Adequate protection can only be achieved by cooperation of different distributed components (Mirkovic et al. 2004).

We think it is possible to find adequate solutions on protection against the Internet attacks, including DDoS attacks, by simulation of present and new attacks and defense mechanisms. It is very important to use right powerful simulation approach and simulation environment which give a researcher an opportunity to comprehensively investigate various modes of attack and defense operation, analyze efficiency of defense (for example, false positives, false negatives; reaction time), develop new methods, etc. *Our goal is to develop the simulation environment which can help investigate the Internet attacks and defense mechanisms (on an*

example of DDoS) and elaborate well-grounded recommendations on the choice of efficient defense mechanisms. In (Kotenko 2005) we suggested the ontology of DDoS attacks and defense mechanisms, the specifications of DDoS and defense agents' team, the formal model of computer network and software prototypes and experiments with them. In (Kotenko et al. 2006) we defined more exactly the used agent-based approach and considered a new simulation environment developed on OMNeT++ INET Framework. In this paper, based on the main ideas considered in (Kotenko 2005; Kotenko et al. 2006) we evolve our approach and simulation environment for comprehensive investigation of DDoS attacks and defense and consider different experiments on investigation various attack defense methods. The rest of the paper is structured as follows. *Section 2* shortly outlines suggested approach for simulation. *Section 3* describes the simulation environment developed. *Section 4* presents the taxonomy of input and output parameters for simulation. *Sections 5 and 6* present the testing methodology for defense mechanisms investigation and the results of experiments fulfilled. *Conclusion* outlines the main results and future work directions.

2. SIMULATION APPROACH

We try to use the agent-oriented approach for simulation of security processes in the Internet. It supposes that the cybernetic counteraction is represented as the interaction of different teams of software agents (softbots) (Kotenko 2005; Kotenko et al. 2006). The aggregated system behavior becomes apparent by means of the local interactions of particular agents in dynamic environment that is defined by the model of computer network. We distinguish at least two agent teams: the attack team and the defense team.

DDoS attacks agents are divided into two classes: "daemon" and "master". Daemons are attack executors. Master coordinates them. On the preliminary stage daemons and master are deployed on available (already compromised) hosts. Then the phase of team establishing takes place. Daemons send to master the messages with information that they are alive and ready to work. Master stores this information about team members and their status. The malefactor sets the mutual team goal – to start the DDoS attack in the given

moment of time. Master receives the attack parameters. Its goal is to send it to all available daemons. Then daemons begin to act. Their local goal is to execute the master instruction. They begin to send the attack packets to the given host in the given mode. Master examines daemons periodically to know that they are workable. Master controls the given attack mode by receiving the replies from daemons. When a daemon does not answer, master decides to change attack parameters. For example, it can send the commands to change the attack intensity to all or particular daemons. Daemons can execute the attack in several modes. This influences on the possibility of defense team to detect and block the attack and to trace and defeat the attack agents. The mode can be specified, for example, by the intensity of packet sending (packets per second) or (and) the method of IP address spoofing.

Defense agents are classified into the following classes: information processing (“sampler”); attack detection (“detector”); filtering (“filter”); investigation (“investigator”). Sampler processes the network packets and creates the model of normal functioning for the given network (in the learning mode). Then in normal mode it analyses and compares the traffic with the model of normal traffic. It picks out the addresses of hosts that do not correspond to the model and sends them to detector. The detector goal is to decide to begin the attack on the basis of sensor and sampler data. Detector sends the list of attack addresses received from sensor or (and) sampler to filter and investigator. Filter blocks traffic on the basis of detector data. Investigator goal is to trace and defeat the attack agents. After receiving the message from detector it examines the received addresses for the presence of attack agents and tries to defeat them.

3. INPUT AND OUTPUT PARAMETERS FOR SIMULATION

We differentiate the input parameters which specify DDoS attack and defense mechanisms.

The scheme of *DDoS attack parameters* is founded on attack taxonomy (Mirkovic et al. 2002). The following criteria were selected (Figure 1):

- *Victim type*. Application, host or network can be chosen. It is necessary to set victim IP address and port.
- *Attack type*. Brute-force (UDP/ICMP flood, smurf/fraggle, etc.) or semantic (TCP SYN, incorrect packets, hard requests).
- *Impact on the victim*. One can choose the disruptive attack (when all daemons attack simultaneously) or degrading (when daemons join the attack one by one). It is easier to detect the attack in the first case.
- *Attack rate dynamics*. It can be constant or variable when the intensity changes in time. The function of attack packets rate is given to daemons. The change

can be increasing (daemons send more and more packets every moment of time) or fluctuating.

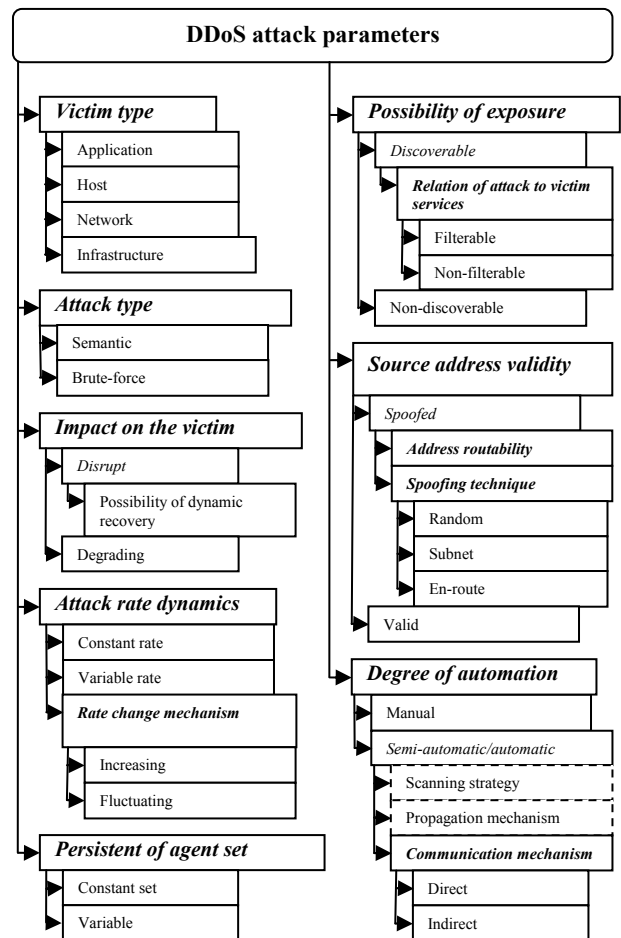


Figure 1. DDoS attack parameters

- *Persistence of agent set*. The set of agents can be persistent (all daemons participate in attack) or variable. In last case master can divide all daemons to several groups which attacks alternately.
- *Possibility of exposure*. The attack can be discovered when it is possible to distinguish the attack packets. We distinguish non-filterable and filterable attacks. Non-filterable attack: the attack packets are formed to be indistinguishable from legitimate. Filterable attack: the attack packets can be discovered by field values, size, etc.
- *Source addresses validity*. Attacker can use the valid (real) or spoofed source address sending the attack packets. One can replace the address to the random chosen or to the address from the same subnet as the daemon. This address can be routable or non-routable.
- *Degree of automation*. Attack can proceed fully automatic after setting the parameters or by the malefactor control. In such a case he (she) can interfere and change one of parameters on all phases of attack. The communication mechanisms between daemons and master can be direct (master knows the

addresses of all daemons) or indirect (agents communicate via a server).

In the experiments that are presented in the paper the following attack parameters were used: Victim type – host (server that provides some service to clients); Attack type – brute-force (UDP flood); 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: constant, random; Degree of automation – semi-automatic with direct communication.

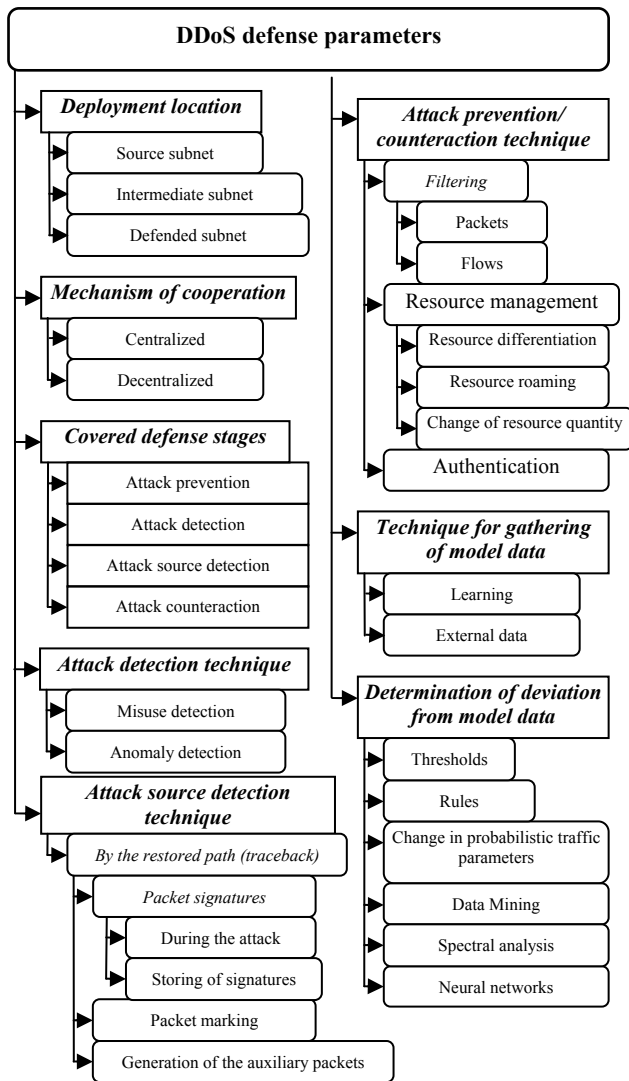


Figure 2. DDoS defense parameters

The scheme of *DDoS defense parameters* is built on the basis of classification proposed by authors. The criteria selected are as follows (Figure 2):

- *Deployment location*: source, intermediate or defended subnets.
- *Mechanism of cooperation*. The mechanism of particular components operation can be centralized or decentralized. In the last case the defense

components are autonomous and can combine their efforts.

- *Covered defense stages*. The stages (mechanisms) the defense method can implement are as follows: (1) attack prevention; (2) attack detection; (3) attack source detection; (4) attack counteraction.
- *Attack detection technique*. There are two types of detection: misuse and anomaly. One chooses one detection method or the set of methods.
- *Attack source detection technique*. Attack source detection (or “traceback”) can be realized by packet signatures, packet marking, generation of auxiliary packets, etc.
- *Attack prevention/counteraction technique*. One can use filtering (of packets or flows), resource management (differentiation, change of quantity, roaming) and authentication.
- *Technique for gathering of model data*. Data can be generated by learning or from external sources.
- *Determination of deviation from model data*. One can use thresholds, rules (for packets and connections), determination of fluctuation in probabilistic traffic parameters, data mining (from traffic statistics), etc. (depending of defense mechanism).

The *output parameters* used to estimate the defense mechanisms are as follows: List of detectable attacks; Time of attack detection (from the start of attack); Time of attack reaction (time from detection to counteraction); Percent of false positives; Percent of false negatives; Computational complexity (quantity of computational resources used), etc.

4. SIMULATION ENVIRONMENT

The approach used for simulation assumes the presence of the following main *simulation environment components* (Figure 3):

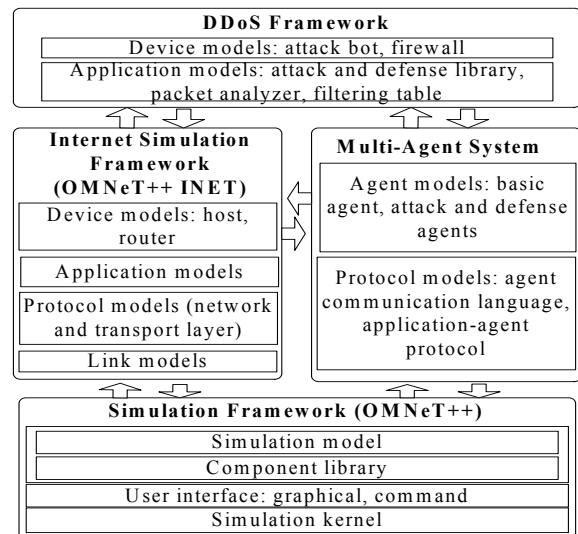


Figure 3. Simulation environment architecture


(1) Basic Simulation Framework (discrete event simulator); (2) Internet Simulation Framework (package for Internet simulation or network simulator); (3) Multi-agent Simulation Framework; (4) DDoS Simulation Framework (the library of attack and defense mechanisms). To choose the Internet simulation tool the comprehensive analysis of the following network simulators was carried out: NS2, OMNeT++ INET Framework, SSF Net, J-Sim INET Framework and some others. We have chosen discrete event simulator OMNeT++ for Basic Simulation Framework and network simulator OMNeT++ INET Framework as Internet Simulation framework (Omnnet++ 2006). On the basis of these tools we have implemented Multi-agent Simulation Framework and DDoS Simulation Framework. *Internet Simulation Framework* is the OMNeT++ *INET Framework* simulation suite which allows the realistic simulation of Internet nodes and protocols. The highest IP simulation abstraction level is the network consisting of IP nodes. IP node can represent router or host. IP node in INET 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: transport (UDP, TCP, including TCP Sockets; routing (MPLS, LDP, RSVP, OSPF-TE) and application (HTTP, Telnet), etc.

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 (FIPA 2006) 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). Agent directory is mandatory only for agent that coordinates other agent in its team. Agent can control the other modules (including DDoS Framework modules) due to messages. There was implemented the agent directory for agents “master” and “detector” that coordinate agent actions in their teams. For daemon there were implemented two transport modules (for communications and attacks). There was implemented filtering table module to let the filter apply filtering rules. Sensor and sampler were provided by the network layer to let them process and collect the network data. This is aimed to create the model of normal traffic.

DDoS Simulation Framework consists of DDoS attack and defense modules and the modules that expand IP node from INET: the filtering table and the packet analyzer. Attack and defense modules are the applications and are deployed on the network layer of IP node. To set the DDoS attack conditions it is necessary to define the corresponding parameters,

including victim type (host), DoS attack type (UDP flood, TCP flood, etc.), attack rate dynamics (function of attack packets sending rate), spoofing technique, etc. Also one needs to set up the defense parameters, including deployment location (defended, intermediate, source subnet), detection technique, model data gathering technique and its parameters (time interval and time shift of data collection), etc. *It is suggested to use open library of different DDoS attacks and defense mechanisms. The main goal is to use the simulation environment for investigation of defense solutions based on state-of-the-art in the Internet protection, and generating valuable recommendations on choosing the best defense.*

The example of *multi-window user interface* of the simulation environment is depicted in Figure 4. At the basic window of visualization (Figure 4, right), a simulated computer network is displayed. The window for simulation management (Figure 4, upper left) allows looking through and changing simulation parameters. It is important that you can see the events which are very valuable for understanding attack and defense mechanisms on the time scale. The time scale is depicted above windows with text description of events. One can see in Figure 4 the events of TCP connection establishing, the actions of sensor, initiation of attack, etc. It is possible to open different windows which characterize functioning (the statistical data) of particular hosts, protocols and agents. For example, at the upper right of Figure 4 the filtering table of one of the hosts is displayed. The applications (including agents, Figure 4, upper middle) are deployed on the hosts (Figure 4, bottom left) in the modeling environment. Their installation is fulfilled by connecting to the modules serving transport and network layers of protocol stack simulated in OMNeT++ INET Framework.

The example of the basic window of visualization (where a *simulated computer network* is displayed) is depicted in Figure 5. The studied network represents a set of hosts and channels. Hosts can fulfill different functionality depending on their parameters or a set of internal modules. The routers are labeled with the oval sign “”. Attack agents are deployed on the hosts marked with red color. Defense agents are located on the hosts marked with green color. Above the colored hosts there are the strings that indicate the corresponding state of deployed agents. Other hosts are standard and generate the standard network traffic. The hosts are connected with the data channels which parameters can be changed.

5. DEFENSE TESTING METHODOLOGY

We are trying to investigate different active and passive defense mechanisms against DDoS attacks. *The main idea of testing methodology* used is to run a large series of experiments for various values of input parameters,

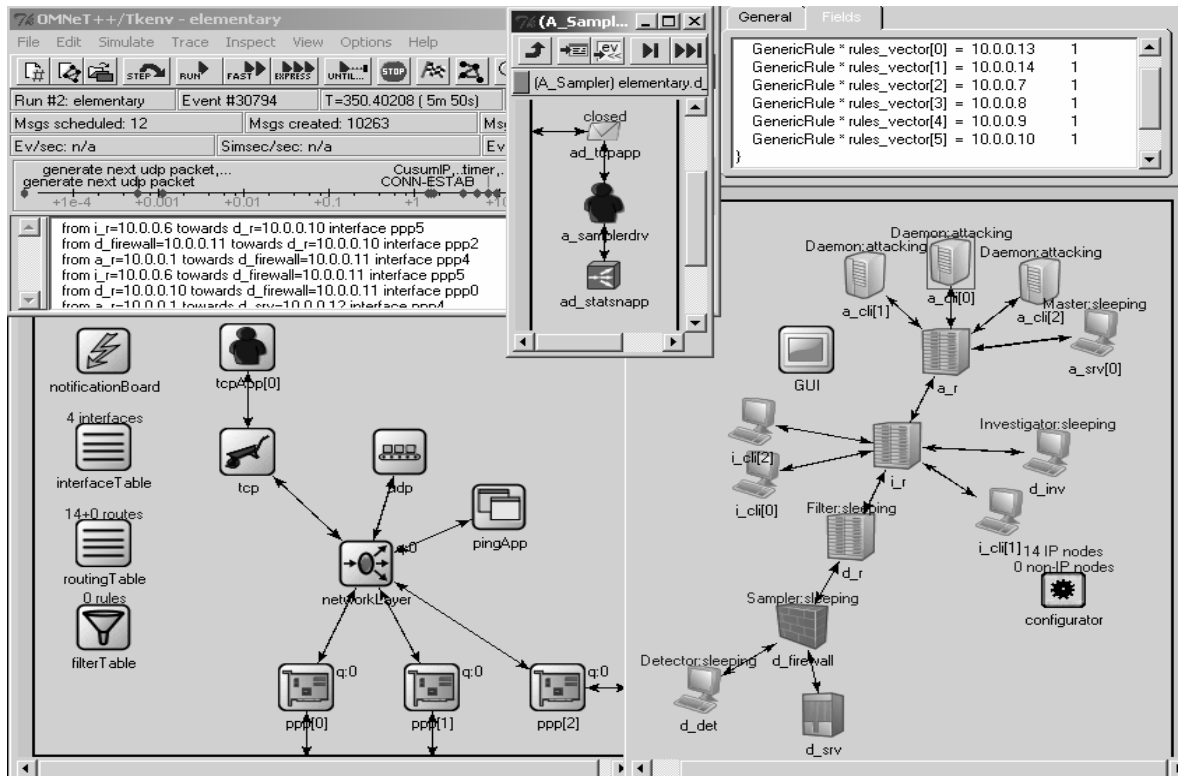


Figure 4. Example of simulation environment user interface

measuring the effectiveness and efficiency parameters of analyzed defense mechanisms and its combination as well as fulfilling its comprehensive analysis. In one paper we can present only limited subset of experiments accomplished. Let us describe at first the main parameters of network topology and channels, network configuration, attacks and defense team configuration used in the experiments described in the paper.

Network topology and channels parameters. 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 $\gamma = 2.25$ (Mahadevan et al. 2005). Routers are connected with the fiber-glass data channels with the following parameters: propagation delay is 1 microsec; datarate is 2.4 Gbit. Other hosts are connected by Ethernet data channels with propagation delay is 0.1 microsec; datarate is 100 Mbit.

Network configuration. 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.

Attack team configuration. Attack team consists of daemons deployed on the standard network hosts that are randomly connected to the routers of the basic network and of agent-master that is deployed on the `a_srv` host. The initial parameters of attack team are: The amount of daemons is 10; the address of master for the team interaction – `a_srv[0]`; the port of master for the team interaction – 2000; the port of daemons for sending the attack packets – 2001; attack target address – `d_srv`; attack target port – 2001; attack start time – 300 seconds; attack rate and IP-address spoofing technique are the input parameters.

Defense team configuration. Detector is deployed on the host `d_det`, sampler – `d_firewall`, filter – `d_r`, investigator – `d_inv`. The base parameters of the defense team are as follows: defended server address – `d_srv`; detector address for team interactions – `d_det`; detector port for team interactions – 2000; Sampler parameters are the input parameters. They depend on the studied defense mechanism.

In the paper we demonstrate the results for three defense methods: Hop Count Filtering (HCF), Source IP address monitoring (SIPM), Bit per Second analysis (BPS).

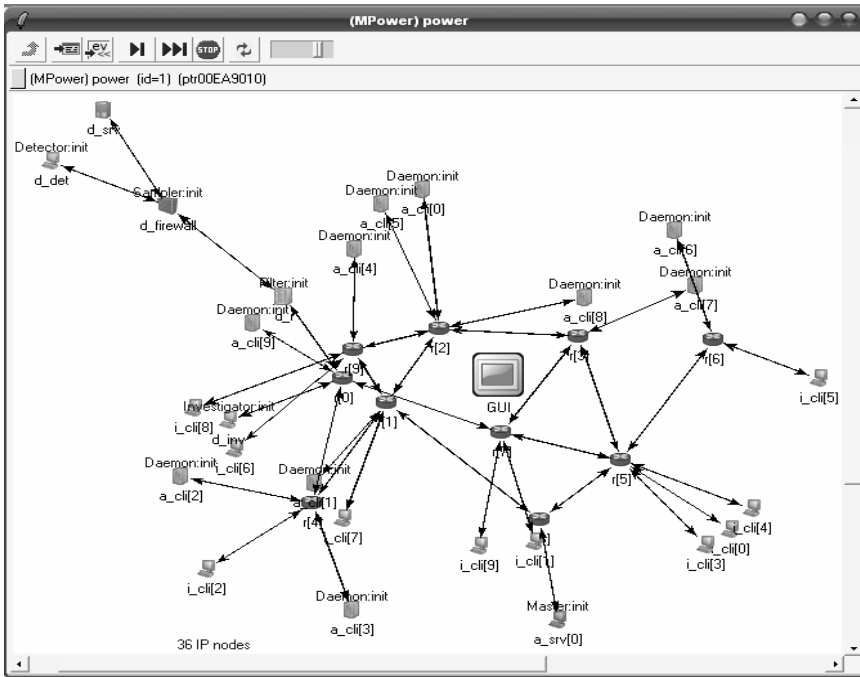


Figure 5. Example of computer network for simulation

Hop count Filtering (HCF) (Jin et al. 2003). It is used the assumption that packets from the same subnet pass through the same hops on the way from sender to receiver. The count of hops is estimated due to packet TTL field. It is decremented on each router. In learning mode, the table based on requests to defended host is created. It consists of IP addresses grouped by their hop count. The system calculates the hop count of incoming packet and compares it with the given value in normal mode. If the hop count differs, the packet is dropped.

Source IP address monitoring (SIPM) (Peng et al. 2003). The assumption is used that in the beginning of attack there are a lot of packets which are sent from new IP addresses and directed to the defended host addresses. In learning mode, the table of legitimate addresses based on clients' requests is created. Both in normal and learning modes the system calculates the amount of new IP addresses for given interval dt with given shift $tshift$. So the amount of new IP addresses is calculated every $tshift$ seconds for previous dt seconds. In learning mode the maximum value (threshold) nIP of new addresses amount is estimated. Then, in normal mode, if the amount of new addresses is lower than threshold, these addresses are stored. If it exceeds the threshold during several intervals (this aggregation is called cumulative sum method, CUSUM), then packets from new addresses are dropped.

Bit per Second traffic analysis (BPS). It is used the assumption that traffic from one IP address should not exceed some critical threshold. In the learning mode it is calculated the amount of transmitted bits per second (BPS) during the given interval for every client requesting defended host. The system determines the

greatest BPS value. In the normal mode if the BPS parameter for some address exceeds the determined threshold then the packets from this host are dropped. This parameter is calculated every $tshift$ seconds on the previous dt seconds.

The following *effectiveness and efficiency parameters of defense mechanisms* were studied: rate of dropped legitimate traffic (false positive rate); rate of admitted attack traffic (false positive rate); attack reaction time. These parameters were studied in dependence on the following *input parameters*: network configuration (the amount of legitimate clients); attack intensity; IP address spoofing technique used in attack; internal parameters of defense mechanisms and their combinations. The *method of spoofing* may be as follows: (1)

Without spoofing ("no spoof") – the real address of host is used; (2) "constant" – an address is randomly chosen, then it is used for sending the attack packets; (3) "random" – with every new attack packet a new address is randomly chosen from the given range of addresses. This range does not intersect with the range of addresses used in the given network; (4) "random real" – with every new attack packet a new address is randomly chosen from the given range of addresses. This range is in the range of addresses used in the given network. The examples of studied *internal parameters of defense mechanisms* are as follows: learning time; the amount of learning examples; interval dt for SIPM and BPS methods; time shift $tshift$ for SIPM and BPS methods; threshold nIP for the amount of new IP-addresses for SIPM method.

6. SIMULATION RESULTS

The experiments showed that the attack intensity and network configuration do not influence essentially on the nature of studied dependences. But BPS method stops to detect the attack with the decrease of attack intensity because it fails to confront big traffic with the given addresses. Network configuration including clients' amount influences on the learning time of methods. It is because the methods are to process data from as most as possible nodes amount. It is necessary to note that for the attack with not spoofed address the investigation agent can defeat the attack agents, therefore the attack intensity decreases. This causes the decrease of attack admissions.

Figure 6 represents the *dependences of false positive and false negative rates on the learning time for BPS method*. One can see that if the learning time increases

then the dropped legitimate traffic rate abruptly decreases for all address spoofing techniques (Figure 6, first graph). But at the same time the amount of admitted attack traffic increases for the random and random real spoofing techniques (for the remaining two methods the amount remains on the same level) (Figure 6, second graph). One could say that, if the learning time was short, the BPS method “closes” the defended system and drops both the legitimate and attack traffics. For the long-term learning this method admits the bigger part of legitimate traffic, but at the same time it is vulnerable to the random and random real address spoofing techniques. The latter is because this method fails to confront the big traffic to the permanently changing address set. The optimal learning time is about 60 seconds. For this time false positive rate is about 10–20% and the false negative rate is from 15 till 40% for the various address spoofing techniques.

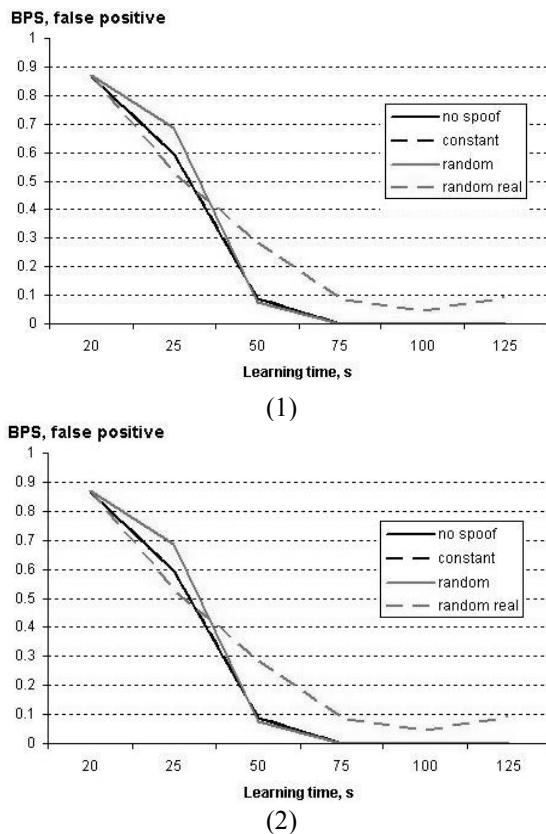


Figure 6. Dependences of false positive and false negative rates on the learning time for BPS method

Figure 7 shows the dependences of false positive and false negative rates on the learning time for HCF method. On the second graph of this figure one can see that the method succeeded to resist only the attack which used the “random real” address spoofing technique (spoofed addresses are from the same network as the legitimate clients). This is because of the following method feature. HCF method can detect only attack packets in which the attacker used the addresses recorded while learning. In the other cases the method

admits as legitimate as well as attack traffic. Method shows poor results and can detect only 20% attack traffic with the average learning time about 50 seconds, but only for the attack with “random real” address spoofing technique.

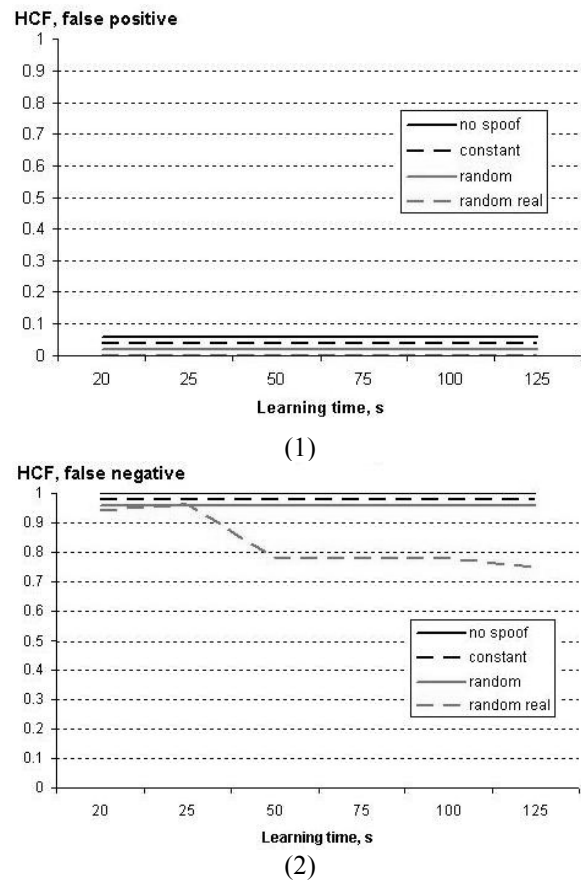
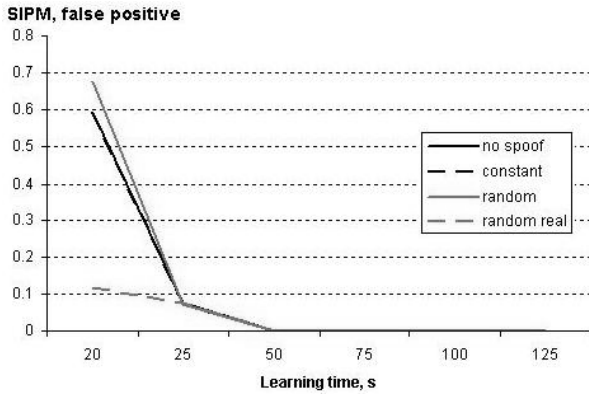
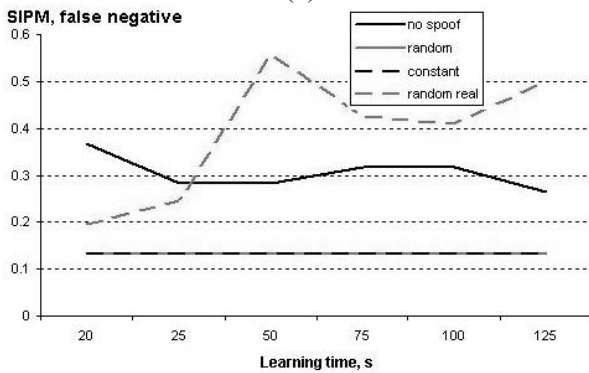


Figure 7. Dependences of false positive and false negative rates on the learning time for HCF method

Figure 8 depicts the dependences of false positive and false negative rates on the learning time for SIPM method. As for BPS, if learning time increases then the dropped legitimate traffic rate abruptly decreases for all address spoofing techniques (Figure 8, first graph). But the learning time does not influence essentially on the admitted attack traffic rate with the exception of random real spoofing technique (Figure 8, second graph). With long-time learning the method “remembers” all client addresses and if attacker uses one of such address it will be admitted. At the same time this method better (in comparison with BPS and HCF) resists to the attacks with all address spoofing techniques. Method can work acceptably with relative short learning time but needs additional tuning: the new IP-addresses threshold and CUSUM threshold are to be chosen correctly. The smaller the first threshold is the more sensitive the method is and the bigger the false positive rate is. Optimal learning time is about 25 seconds, where false positive rate is less than 10% and false negative rate is from 15 to 30%.

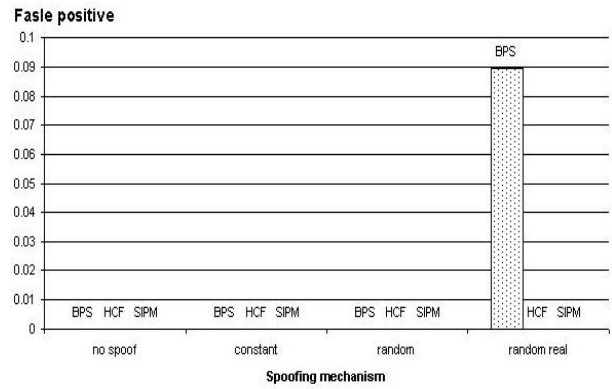


(1)

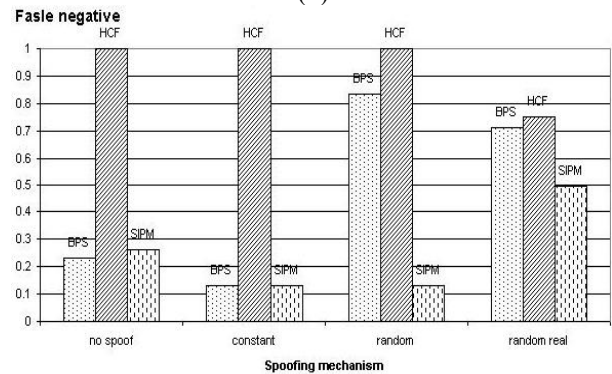


(2)

Figure 8. Dependences of false positive and false negative rates on the learning time for SIPM method



(1)



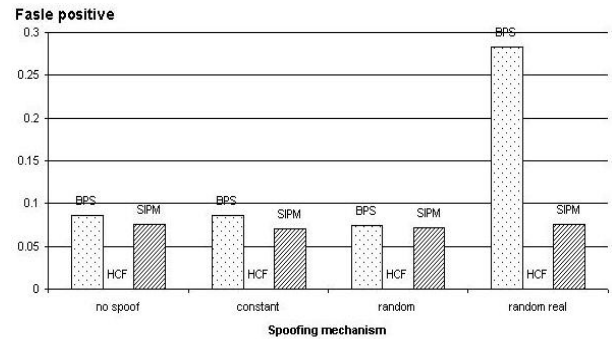
(2)

Figure 9. Dependences of false positive and false negative rates on the learning time for BPS, HCF, SIPM

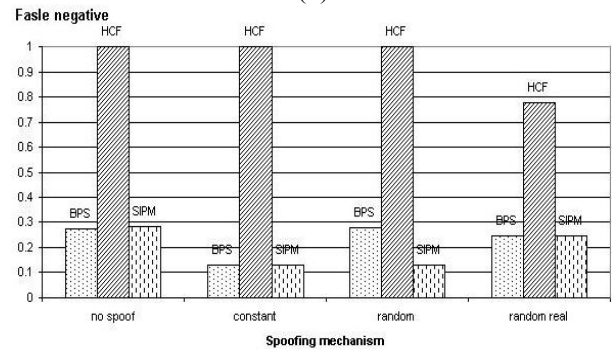
Figure 9 shows the dependences of false positive and false negative rates on the learning time for BPS, HCF and SIPM for the maximum learning time (125 seconds). One can see (Figure 9, first graph) that all methods except BPS have no false positives and legitimate traffic is completely admitted. But the BPS false positive rate is small and appears only with attack using “random real” spoofing technique. The smallest attack admission rate (Figure 9, second graph) is provided by SIPM and BPS methods. However the latter is more sensible to attack intensity and to attack spoofing technique.

Figure 10 depicts the dependences of false positive and false negative rates on different address spoofing techniques for BPS, HCF and SIPM for the optimal learning times. SIPM method provides the smallest false positive rate (less than 8%). False negative rate for BPS and SIPM are roughly the same for different spoofing techniques: from 15 till 30%. HCF has unsatisfactory false negative rate.

Figure 11 shows the dependence of reaction time on various address spoofing techniques for BPS, HCF and SIPM methods. Time is 0 seconds if the method did not worked: there were neither false positive nor false negative. The reaction time is about 40 seconds for all methods except HCF.



(1)



(2)

Figure 10. Dependences of false positive and false negative rates on different address spoofing techniques for BPS, HCF and SIPM for the optimal learning times

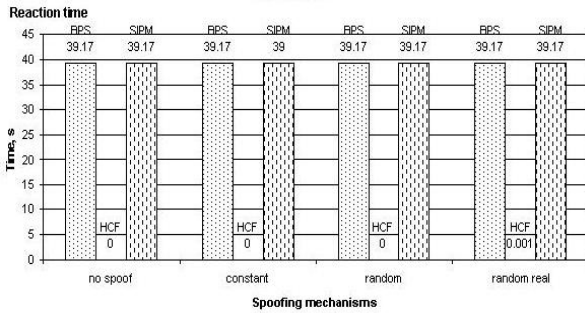


Figure 11. Dependence of reaction time on various spoofing techniques for BPS, HCF and SIPM methods

This amount of time is spent mainly on interaction between sampler, detector and filter and on applying filtering rules. HCF method allows only local filtering (on the same host where HCF method was deployed). This feature causes such small attack reaction time on attack with “random real” address spoofing technique.

Figure 12 represents the dependence of false positive and false negative rates on address spoofing technique for BPS, HCF and SIPM defense methods working together (for the maximum learning time 125 seconds). One can see that such regime provides smaller rates of dropped legitimate traffic and admitted attack traffic than for each of methods separately. This is because the methods discover different parts both legitimate and attack traffic from the whole traffic. That is together these methods “compensate” each other, and their joint use is more effective than independent one.

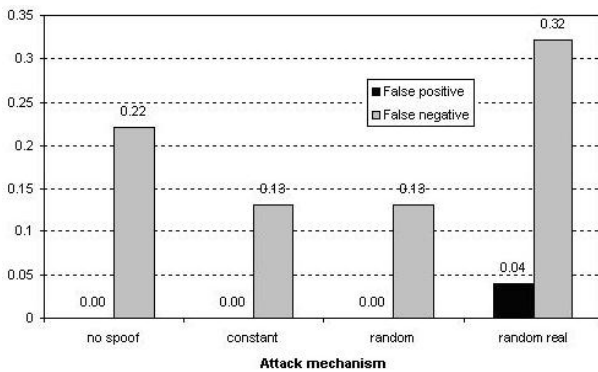
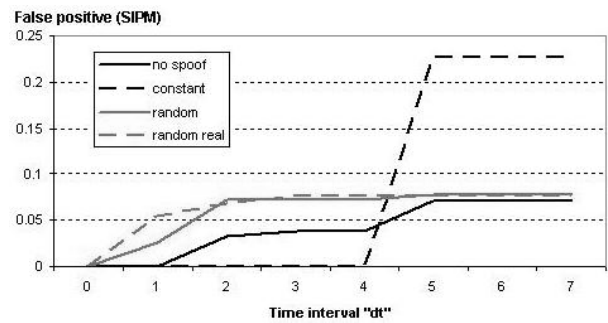


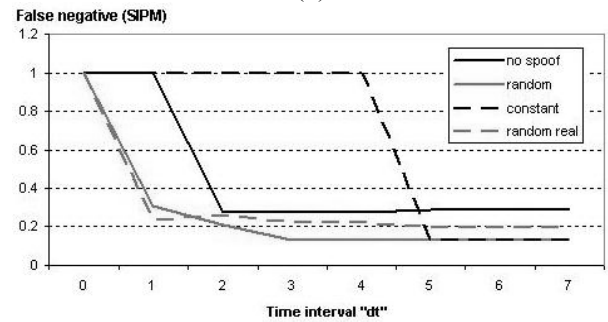
Figure 12. Dependence of reaction time on various spoofing techniques for BPS, HCF and SIPM methods

The dependence of false positive and false negative on the internal SIPM parameters was also studied: dt — interval for data collection; $tshift$ — interval time shift; nIP — threshold for the amount of new IP-addresses; Threshold for nIP exceeding was 1. Interval for data collection (dt) influences on the possibility of detecting attacks with various impact on the victim and rate dynamics. The bigger interval allows to resist to degrading DDoS attack when the packets with new IP-addresses arrive gradually. The smaller interval ensures quicker reaction on attack, since it is decided if there is

an attack at the end of each interval. In the case of disrupt attack (that is represented in this paper) interval can be short enough. Its increase influences only on reaction time and decrease leads to the drastic loss of attack sensibility (Figure 13). With $dt < 1$ the false positive rate increases (Figure 13, first graph). At the same time the biggest amount of attack traffic is admitted, but the false negative rate decreases (Figure 8, second graph). With increasing dt the false positive rate stays the same for all address spoofing technique, except “constant” while false negative rate sudden decreases and then is not changed. The sensibility can be compensated by the short time interval shift. The most acceptable dt is about 5 seconds.



(1)



(2)

Figure 13. Dependence of false positive and false negative rates on the interval dt of data collection for SIPM

Time shift $tshift$ influences on reaction time. The more it is the later reaction comes. But small shift demands more computational expenses.

The threshold for the new IP-addresses nIP influences on the attack admission rate most of all. If the learning was held for the maximum time and mechanism learned all constant “clients” of defended system, then the threshold can be very little. In this case the amount of attack admissions is minimal: the packets with new IP-addresses will be most likely the attack packets. However it is usually hard to learn all possible clients. It is interesting that for the optimal learning time, threshold nIP influences on false positive rate as follows. With little threshold nIP one can see more false positive, than with the achievement of some value and above it (Figure 14). This is because mechanism does

not have enough records of legitimate addresses and high threshold can not influence on attack detection. The optimal threshold nIP is 6.

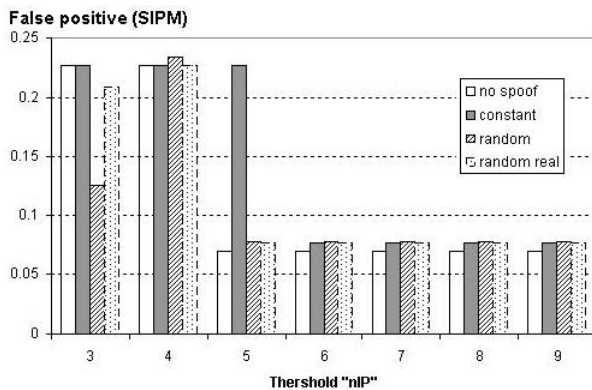


Figure 14. Dependence of false positive rate from nIP threshold for SIPM

7. CONCLUSION

The main results of the work we described in the paper consist in implementing the agent-based simulation environment intended for simulation of DDoS attacks and defense. The goal of the paper was to show the possibilities of the simulation environment developed. The agent-based simulation environment allows to investigate attacks and defense methods and to provide valuable recommendations on choosing the best defense. The environment allows simulating a wide spectrum of DDoS attacks and defense mechanisms. Various experiments with this environment have been fulfilled. In the paper we demonstrated by agent-based simulation the effectiveness parameters for three different defense methods: Hop Count Filtering (HCF), Source IP address monitoring (SIPM), Bit per Second analysis (BPS). The results of these experiments have been outlined in the paper.

Future work is concerned with improving the functionality of the simulation environment and investigating new defense mechanisms, including cooperative defense methods implemented in COSSACK, Perimeter-based DDoS defense, DefCOM, Gateway-based, ACC pushback, MbSQD, SOS, tIP router architecture, etc. (Mirkovic et al. 2004; Xiang et al. 2004; Yuan et al. 2005).

8. ACKNOWLEDGEMENT

This research is being supported by grant of Russian Foundation of Basic Research, program of fundamental research of the Department for Informational Technologies and Computation Systems of the Russian Academy of Sciences (contract No 3.2/03), Russian Science Support Foundation and partly funded by the EU as part of the POSITIF project (contract No. IST-2002-002314) and RE-TRUST (contract No.021186-2).

REFERENCES

- Chen, S. and Q.Song. 2005. "Perimeter-Based Defense against High Bandwidth DDoS Attacks". *IEEE Transactions on Parallel and Distributed Systems*, Vol.16, No.7.
- FIPA. <http://www.fipa.org>
- Jin, C.; H. Wang; K.G. Shin. 2003. "Hop-count filtering: An effective defense against spoofed DDoS traffic". *Proceedings of the 10th ACM Conference on Computer and Communications Security*.
- Kotenko, I. 2005. "Agent-Based Modeling and Simulation of Cyber-Warfare between Malefactors and Security Agents in Internet". *19th European Simulation Multiconference "Simulation in wider Europe"*.
- Kotenko, I. and A. Ulanov. 2006. "Agent-based Simulation of Distributed Defense against Computer Network Attacks". *20th European Simulation Multiconference "Simulation in wider Europe"*.
- Mahadevan, P.; D.Krioukov; M.Fomenkov; B.Huffaker; X.Dimitropoulos; K.Claffy; A.Vahdat. 2005. "Lessons from Three Views of the Internet Topology". *Cooperative Association for Internet Data Analysis (CAIDA)*.
- Mirkovic, J.; J. Martin; P. Reiher. 2002. "A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms". *Technical report #020018*. University of California, Los Angeles.
- Mirkovic, J.; S. Dietrich; D. Dittrich; P. Reiher. 2004. "Internet Denial of Service: Attack and Defense Mechanisms". Prentice Hall PTR.
- OMNeT++. <http://www.omnetpp.org/>
- Peng, T.; L. Christopher; R. Kotagiri. 2003. "Protection from Distributed Denial of Service Attack Using History-based IP Filtering". *IEEE International Conference on Communications*.
- Yuan, J. and K.Mills. 2005. "Monitoring the Macroscopic Effect of DDoS Flooding Attacks". *IEEE Transactions on Dependable and Secure Computing*, Vol.2, No.4.
- Xiang, Y.; W. Zhou; M. Chowdhury. 2004. "A Survey of Active and Passive Defence Mechanisms against DDoS Attacks". *Technical Report, TR C04/02*, School of Information Technology, Deakin University, Australia.

BIOGRAPHY



IGOR KOTENKO graduated with honors from St.Petersburg Academy of Space Engineering and St.Petersburg Signal Academy. He obtained the Ph.D. degree in 1990 and the National degree of Doctor of Engineering Science in 1999.

He is Professor of computer science and a head of computer security research group in St. Petersburg Institute for Informatics and Automation. His Web-page can be found at <http://comsec.spb.ru/kotenko/>.



ALEXANDER ULANOV graduated from St. Petersburg State Politechnical University (2004), received his master's degree (2004) in the area "System analysis and control". He is PhD student in the field of agent-based simulation of

computer network attacks and defence. His Web-page can be found at <http://comsec.spb.ru/ulanov/>.