Investigation of DDoS Attacks by Hybrid Simulation

Yana Bekeneva, Konstantin Borisenko, Andrey Shorov, Igor Kotenko

Investigation of DDoS Attacks by Hybrid Simulation

Yana Bekeneva [1], Konstantin Borisenko [2], Andrey Shorov [3], Igor Kotenko [4,5]

[1,2,3] Department of Computer Science and Engineering
Saint-Petersburg Electrotechnical University "LETI"
Professora Popova str. 5, Saint-Petersburg, Russia
[1]yana.barc@mail.ru [2] borisenkoforleti@mail.ru [3]ashxz@mail.ru
[4] St. Petersburg Institute for Informatics and Automation
14th Liniya, 39, Saint-Petersburg, Russia
ivkote@comsec.spb.ru
[5] St. Petersburg National Research University of Information Technologies,
Mechanics and Optics, 49, Kronverkskiy prospekt, Saint-Petersburg, Russia

Abstract. At present protection against distributed attacks of the type "denial of service" (DDoS) is one of the important tasks. The paper considers a simulation environment for DDoS attacks of different types using the combination of a simulation approach and real software-hardware testbeds. In the paper we briefly describe the system architecture and a series of experiments for DDoS attack simulation on transport and application levels. The experimental results are provided, and the analysis of these results is performed.

Keywords: network security, DDoS attacks, simulation, Flooding

1 Introduction

The main goal of DDoS attacks is to make a network resource unavailable to its users. Every year there is an increase in the number DDoS attacks, their power and complexity and hence the harm they can do. According to the report of the Arbor Network [1], more than 60% of companies that have been questioned detected more than 10 DDoS attacks per month in 2014. By the year 2015 the main victims of such attacks are not only servers of different companies and Internet providers, but also their clients. New types of DDoS attacks appear and one of destroyable type of attacks is so called Reflection attacks. These attacks use servers to reflect and amplify the malicious traffic. Nowadays Network Time Protocol (NTP) Reflection attacks using NTP servers became very popular. This means that it is necessary to develop new efficient protection mechanisms against DDoS attacks. Development of new protection systems and testing of real networks for stability to DDoS attacks requires significant hardware, temporal, and financial costs. For this reason in order to perform such experiments we suggest using computer simulation techniques.

As against of our previous papers [2, 3] we would like to present the system, having included the possibility of using real nodes connected to a virtual network. Such an approach has made it possible to significantly increase the adequacy of DDoS attack simulation on transport and application levels. The system has been verified for a real

network. The paper is devoted to the experiments for simulating different types of DDoS attacks in the developed environment. The authors have conducted the experiments of simulating DDoS attacks on transport and application levels. We have simulated the attacks using the protocols of TCP (for SYN-flooding) and UDP (for Chargen and Echo attacks). Attack simulation on an application level has been performed. Section 2 discusses related work. Section 3 specifies formal models of main components. In section 4 we briefly represent the architecture and implementation. Sections 5 and 6 consider examples of experiments. Section 7 and Conclusion outline application of the developed system, main results and further research.

2 Relevant works

Nowadays a large number of researches and companies are looking for the most efficient ways of service protection against DDoS attacks [3]. In [4] one can get acquainted with a system that uses Spirent Test Center [5] as a device generating traffic. This work contains templates for setting IP-address configurations and attack scenarios. In order to deal with similar tasks it is also possible to use the environment produced in 2014 by the company MazeBolt's Team [6]. Using this environment we can generate different types of DDoS attacks of the power up to 20 000 Mbit/sec. [7] considers a system developed by Ixia [8]. The authors point out that the system makes it possible to provide protection against DDoS attacks of the power of 70 Gbits/sec. in real time. ViSe [9] is a system for simulating the main most widespread attacks (40 different scenarios). In [2] we can read about a DDoS attack simulation system, which allows us to simulate a network with different behavior of clients inside it.

The system introduced in our paper allows us to construct virtual networks with a high degree of adequacy. Furthermore, the system makes it possible to introduce any known protection mechanism or the one created by the user. Protection mechanisms can be architecture-dependent, which also allows us to conduct experiments in the way most close to a real network. An important advantage is the possibility of connecting real nodes to a virtual network, which will make it possible to improve the accuracy of the experiments and also to test various settings and types of servers.

3 Specification of simulation components

In order to understand the structure of the developed simulation components, it is necessary to specify the parameters of their models. The formal model of the network is defined as follows: $Network = \langle NetConf, Router_{1..q}, Host_{1..n}, Server_{1..m} \rangle$, where *NetConf* — network configuration, *Router* — routers, *Host* — hosts, *Server* — servers.

Network configuration $NetConf = \langle IPStart, ExtIP, ServPath, NetPar \rangle$, where *IPStart* — initial IP-address for assignment to hosts, for example, if *IPStart* = "1.0.0.1", and a network consists of 20 hosts (personal computers (PCs), routers, servers), then the IP address of the last host is 1.0.0.20; *ExtIP* — IP address of the external server; *ServPath* — the path to the virtual server, which will be a plug (it reflects the

position of the real server) for the virtual network; *NetPar* — on the basis of the above components each network host will receive its interface settings and routing tables.

Model of the router $Router_k = \langle NetPar, NPcap, Def_{0..n}, ExtDevN_{0..q}, DelConf \rangle$, where *NetPar* — router's parameters obtained during initialization of Netconf; *NPcap* — the number of modules to account the traffic flow; *Def* — a protection algorithm running on the router, this algorithm is implemented in software, the router may use any number of algorithms; *ExtDevN* — the number of interfaces connected to the external network, required for routers which contain an external host in their local network; *DelConf* — configuration of packets delay, it is used that the virtual network would have the characteristics of the real network. Model of virtual clients $Host_k = \langle NetPar, DDoSApp_{0..1} \rangle$, where *NetPar* — client's settings received under initialization of *Netconf*; *DDoSApp* — an application that will perform the attack for a given scenario. Model of attacks against the server includes the following options: $DDoSApp = \langle VictimPath, AType, Lvl, Dport, DeltaT, AStart, AEnd, MaxP, SpecialP \rangle$, where *VictimPath* — a path to the victim server (to empower attacks against several servers); *Atype* — type of the attack against the server: 1 — HTTP Flooding, 2–7 — different variants of TCP Flooding (SYN, SYN-ACK, RST, UDP, etc.); *Lvl* — the percent of clients involved in the attack;; *Dport* — destination port, it is defined in a random manner; *DeltaT* — the time between packets (in the case of HTTP Flooding it is the time between sessions); *AStart, AEnd* — start and end time of the attack; various conditions for this interval can be set; *MaxP* — the maximum number of packet sendings (in the case of HTTP attacks — sessions); *SpecialP* — specific attack parameters, i.e. for HTTP it is a string of the HTTP query, for TCP — the type of the sender's IP address spoofing. The developed models allow connecting any number of external servers to the virtual network. In the router it is also possible to configure the attack scenarios and the DDoS protection methods.

4 Architecture and implementation of a simulation system

The architecture of the developed system can be represented as a set of several components (Fig. 1). On the first level of the component hierarchy we can find a discrete-event simulation system OMNeT++ [10]. The second level of hierarchy is represented by a model of a computer network. In order to adjust the network and set package switching the INET library is used [11]. The library ReaSE [12] has been completed for topology settings. The authors ceased to support it in 2011 and therefore it was renewed for working with the version OMNeT++ 4.5. The third level contains the models of hosts and routers that are included in the network. The server model is related to both this level and the next one. The fourth level contains the following models developed: an application for performing attacks according to the given scenario; traffic accounting modules; a container for protection algorithms, which can be installed at different network nodes. The models and architecture that have been created were used for developing a hybrid simulation system. In order to provide virtualization of operating systems and computational resources, VirtualBox и Vmware [13, 14] have been applied. The possibilities for risk analysis [15, 16] can be added.

Examples of settings for created components are shown below. Fig. 2 depicts an example of the topology which includes models *Network, NetConf, NetPar*. This topology contains a Web-server, routers, gateways and hosts. Every node can be configured by user any time. Example of settings for NetConf is shown in Fig. 3.

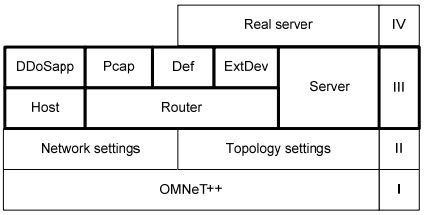

Fig. 1. Common architecture of modeling system

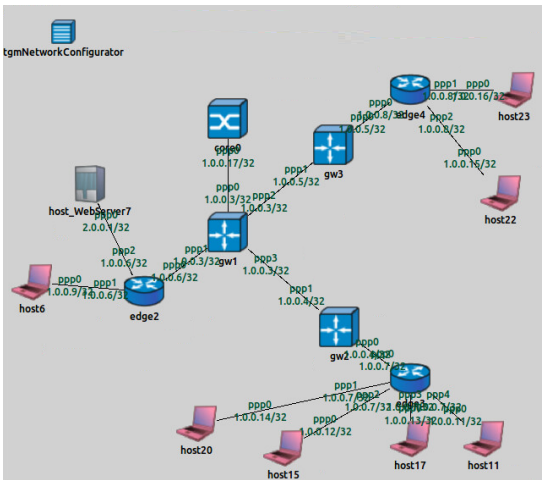

Fig. 2. Example of the created topology

```
#interface and route settings
**.ExtServerIP = "2.0.0.1"      #ip of real server
**.ExtServerNedPath = "InetVerific.host_WebServer7.ppp0"
**.IniIp = "1.0.0.2"
```

Fig. 3. An example of settings for the network configuration model

It should be noted that the range of addresses, the real IP address of the server belongs to, must not overlap with IP addresses of virtual PC. This is to ensure that the server's responses are forwarded to computer of the virtual network. For the experiments we used a computer with the following characteristics: processor is Intel Core i7-3770, 3,4 Ghz, 8 cores; RAM is 15,6 GB DDR3 1600 Mhz; Operating system is Ubuntu 14.04 64-bit. The developed system can also be deployed on the Windows computers. The system currently operates in a single-threaded simulation mode. More than 200 virtual desktops are used, they can attack with the delay of 10 ms. During attack simulations the capacity of the core is 50%.

5 Experiments with attacks on transport level

The authors have conducted a series of 10 experiments devoted to simulating an attack of the type SYN-Flooding In order to successfully perform a SYN-flooding attack, SYN cookies were switched off on a server. The attack involves 200 hosts generating packages with a frequency of 2 packages per second. 3 hosts are attacking throughout the whole experimental time; the rest are included in an attack in a distributed way from the 10th to the 35th second of the experiment and stop package generation in the interval from the 50th to the 70th seconds. Fig. 4 shows SYN packages sent to a server from a virtual network; the line shows the mean value for 10 experiments and errors are also demonstrated. Server responses are represented in a similar way without repeated sending of SYN-ACK for semiopen connections that have already been established.

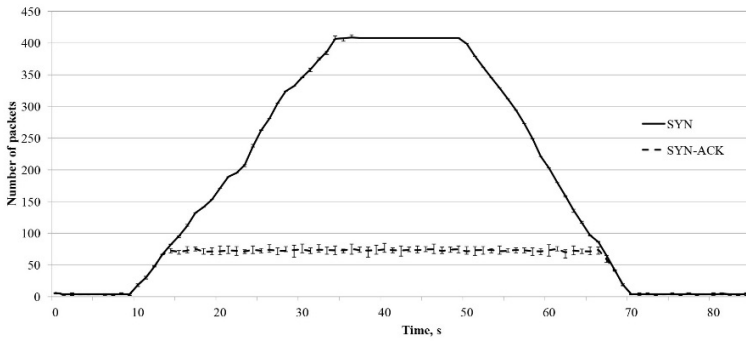

Fig. 4. Plot of the traffic log for SYN Flooding against the server

Only 3 clients applied to the server prior to the 10th second. At the beginning of an attack, on the 10th second, the number of server applications increases because more and more virtual clients are beginning to take part in the attack. Till the 14th second the server manages to process all messages and after that the TCP-stack is overfilled and the server is unable to deal with an increasing flow of applications. Then, beginning from the 50th second and ending with the 70th second virtual clients are leaving the attack. 3 clients are applying to the server again from the 70th on the 88th second. Simulation was stopped at the 88th second. In the period from the 14th second till the 68th second the server did not respond to SYN-packages.

The experiments devoted to attack simulation and based on the UDP-protocol used the network consisting of 100 computers. The attacking packages were sent at a rate of 2 packages per second. The nodes started generation of malicious packages from the 1st to the 20th seconds and terminated the attack from the 40th to the 60th seconds. The Chargen attack is performed by sending a large number of packages to the UDP-port of the 19th victim computer. This causes the server to generate responses consisting of a random set of symbols. Thus, it is possible to reboot the server and reduce its efficiency. The solid line in Fig. 5 shows malicious traffic arriving at an attacking server. Prior to the 20th second we can notice traffic rise, which remains invariable till the 40th second. Then, the number of malicious packages starts to de-

crease. The dashed line shows server responses (chargen). After sending some number of responses the server becomes inactive and instead of UDP-packages generates ICMP-messages about the server unavailability. ICMP-messages are shown in the plot by the dot-and-dash line. Traffic resending is reached when a server being attacked has to send return packages to the UDP-port of the 19th sender, which generates return chargen packages to the server.

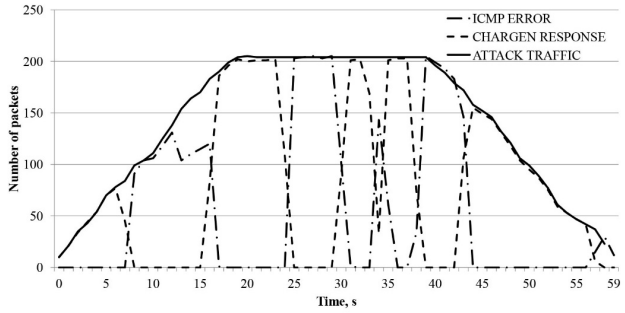

Fig. 5. Server traffic for a Chargen attack

Fig. 6 shows server traffic during an attack with resending. The solid line indicates incoming malicious packages coming from different UDP-ports except the 19th one. The dot line shows incoming UDP-packages sent from the 19th port of attacking nodes, i.e. obtained during resending. The dashed line (short dashes) shows chargen server responses. The curve of chargen responses almost completely coincides with the curve of resent packages. The line with long dashes shows ICMP-messages of the server telling us that the server is currently unavailable.

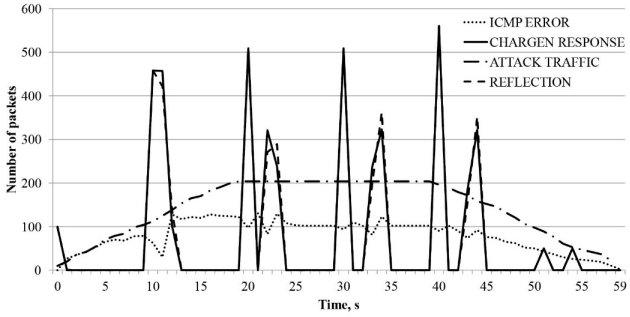

Fig. 6. Server traffic for a Chargen attack with resending

Attack Echo is carried out by sending a large number of packets on the UDP port 7 of a victim computer. This causes the server to generate responses that mimic the incoming packet. As Chargen, this attack leads to overload of the server, and reduces its productivity.

Fig. 7 outlines by the solid line the malicious traffic directed to the target server. The traffic increases up to 20 second, then it remains unchanged up to 40 second, and

further the number of malicious packets is reduced. The dashed line shows the server's response (echo). After sending a certain number of responses the server becomes inactive and generates ICMP packets (instead of UDP ones) that the service is not available. The graph demonstrates the ICMP packets by the dotted line. Bounced traffic is generated when the attacked server needs to send response packets to the UDP port 7 of the sender, which in turn generates echo response packets to the server. However, the implementation of such an attack cannot achieve infinite loop, as the server, as in the previous case, at some time becomes inactive and does not respond to incoming traffic.

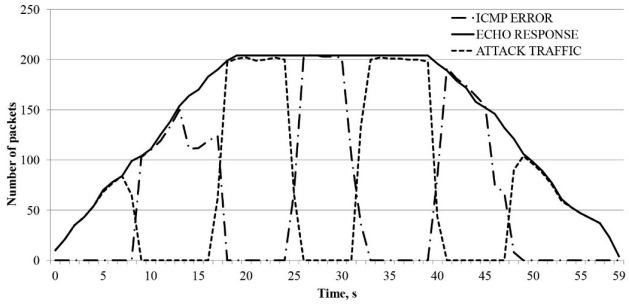

Fig. 7. Server traffic under an Echo attack

Fig. 8 depicts by solid line the incoming UDP packets to the server, including bounced traffic. The dotted line shows Echo responses of the server. It can be seen that when the server responds to Echo requests the incoming traffic includes balanced Echo messages from attacking hosts. Dotted line represents the ICMP packets, indicating that the server is currently inactive.

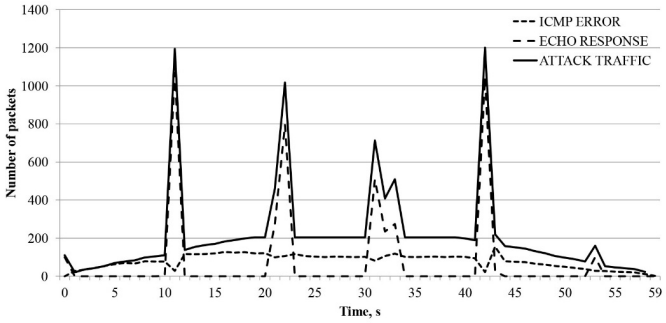

Fig. 8. Server traffic under an Echo attack with bounced traffic

6 Experiments with attacks on application level

An HTTP-Flooding attack is reached by sending a large number of GET queries to the port 80 and as a result the server becomes incapable of processing other queries. A series of 10 experiments has been conducted and 200 hosts are involved in the attack.

Each host initiates a TCP-connection with an HTTP-query at a rate of 2 packages per second. The queries are processed by the server using the PHP script, which increases the server load. Due to increasing load the server response time increases with the increase of the number of hosts. The host studied is participating in the attack during the whole experiment; other hosts are included in the attack in a distributed way from the 10th to the 40th seconds and stop package generation from the 70th to the 90th second. The plot (Fig. 9) shows the dependence of session duration from its number. The line shows the mean value for 10 experiments and the errors are also indicated. As can be seen from the plot, the server load increases during an attack, which makes the query processing time longer.

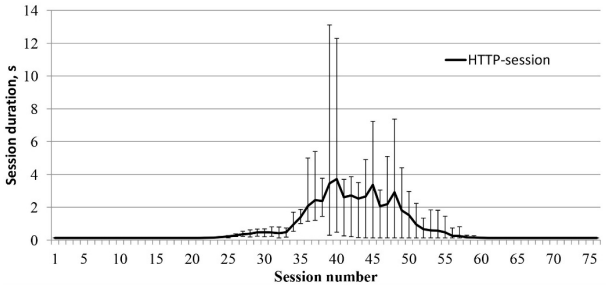

Fig. 9. Traffic log plot for HTTP Flooding against the server

Next NTP Reflection Attack was simulated. It is based on an NTP vulnerability consisting in the fact that if the packet with the request on the latest hosts, which asked for his time, was sent, the NTP-server will send the list of these hosts.

Attack of the type NTP Reflection Attack is implemented the following way. The attacking nodes send the queries get_monlist to an NTP-server; at the same time the source address is replaced with the address of a victim computer. In response to this command the server sends a package containing 600 IP-addresses applying to it earlier. We can gain 500 times attack increase. Consider a scenario when 30% of computers included in a network are participating in the attack (Fig.10).

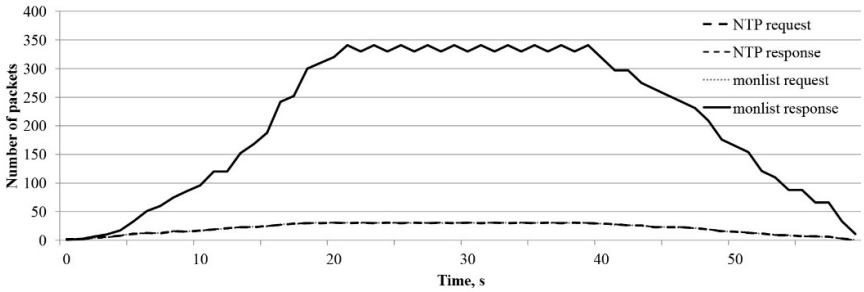

Fig. 10. Server traffic for an NTP-attack (30% of network computers are in an attack)

Each of the attacking nodes generated 2 packages per second. One of the packages was a legitimate query to an NTP-server, i.e. a query for time synchronization. Ad-

dress substitution has not been used. The second package contained the command get_monlist, and the IP-address of the source was replaced with the IP-address of a server being attacked. The nodes started sending packages in a distributed way from the 1st to the 20th seconds and terminated the attack from the 40th to the 60th seconds. The plot (Fig. 10) shows time dependencies of the number of queries and responses. The curves of legitimate queries and NTP responses and also get_monlist queries coincide completely and are shown by the solid line. Server responses to the command get_monlist are shown by the dashed line.

7 Discussion

There are many systems that can simulate DDoS attacks [3, 4, 5, 6, 7, 8, 9, etc.], including systems that generate real traffic for malicious attacks against legitimate sites. But they all have drawbacks that make it difficult to use them. The system presented in [5] cannot be applied to create defense mechanisms. The systems [6, 8] can provide protection against the attacks with limited power; moreover, these systems are very expensive. ViSe [9] requires a lot of hard disc space as well as a rather large number of settings for starting and repeating the tests. Our system can be considered not only as a packets generator for various types of DDoS attacks, but also as a research laboratory to investigate existing DDoS attacks and methods of protection against them. The main advantage of the developed system is that it allows creating and testing new distributed architecture specific defense mechanisms against DDoS attacks. This is achieved by the fact that by using this laboratory it is possible to create a large-scale computer networks consisting of thousands of nodes, generating as attacks as well as the legitimate traffic. The ability to use real hosts allows to work not only with the web server, but with any network services without the need for simulation, for example, with cloud computing systems.

The ability to use not emulated, but real software is very important to simulate DDoS attacks with the highest power, as different software can react differently to events taking place in its environment. For example, if the attacked host is an actual web server, it can have different hardware configurations, settings, software installed on it, so that the reaction of one attacked host may differ significantly from others. The virtual (simulated) part of the system also gives the possibilities to configure parameters of network interfaces for clients and routers (connection speed, performance of routers, etc.). Experiments have shown that the developed system allows quite accurate simulation of the behavior of attacked web servers. For example, in the experiments on SYN Flooding the server stops processing all SYN packets from clients. This corresponds to the official CERT document describing SYN Flooding [17].

8 Conclusion

In the present paper we have introduced the experiments devoted to simulation of different types of DDoS-attacks. The experiments were conducted in the modeling environment developed by the authors with the possibility of connecting real nodes. In the paper we have provided the experiments for attack simulation at transport and

application levels. The experiments have shown that the developed system performs adequate simulation of the processes occurring in computer networks. The behavior of a server being attacked when it is overloaded corresponds to a real server. All the results and plots obtained during simulation in the developed environment are almost identical to the results that could have been achieved in a real network. The developed system can be used in the field of computer security for creating and testing protection mechanisms against DDoS attacks. In future we plan to use this environment for developing new protection techniques against DDoS attacks.

Acknowledgements. This research is being supported by grants of RFBR (projects 13-01-00843, 13-07-13159, 14-07-00697 and 14-07-00417), state project "Organization of scientific research" of the main part of the state plan of the Board of Education of Russia, project part of the state plan of the Board of Education of Russia (task # 2.136.2014/K) as well as by Government of the Russian Federation, Grant 074-U01.

References

1. Worldwide Infrastructure Security Report. ARBOR Networks reports 2014, http://www.arbornetworks.com/resources/infrastructure-security-report (2014)
2. Konovalov, A., Kotenko, I., Shorov, A.: Simulation-Based Study of Botnets and Defense Mechanisms against Them. Journal of Computer and Systems Sciences International, vol.52, issue 1, pp.43-65. Pleiades Publishing, Ltd. (2013)
3. Kotenko, I., Konovalov, A., Shorov, A.: Agent-based Modeling and Simulation of Botnets and Botnet Defense. Conference on Cyber Conflict. Proceedings 2010. CCD COE Publications. Tallinn, Estonia (2010)
4. Wang, J., Phan, R., Whitley, J., and Parish, D.: Advanced DDoS Attacks Traffic Simulation with a Test Center Platform. International Journal for Information Security Research (IJISR), Volume 1, Issue 4 (2011)
5. Spirent TestCenter, http://www.spirent.com/Ethernet_Testing/Software/TestCenter
6. MazeBolt developer, https://mazebolt.com
7. Butler, B.: Interop network squares off against controlled 70G bit/sec DDoS attack, http://www.networkworld.com/article/2166091/data-center/interop-network-squares-off-against-controlled-70g-bit-sec-ddos-attack.html (2013)
8. About Ixia, http://www.ixiacom.com/about-us/company
9. Årnes, A., Haas, P., Vigna, G., Kemmerer, R. A.: Using a virtual security testbed for digital forensic reconstruction. DIMVA 2006, pp.144-163 Springer-Verlag, France (2006)
10. OMNeT++ Discrete Event System Simulator, http://www.omnetpp.org/intro
11. INET Framework, http://inet.omnetpp.org/
12. ReaSE, developer web-site, https://i72projekte.tm.uka.de/trac/ReaSE
13. VirtualBox, developer site, https://www.virtualbox.org/wiki/Technical_documentation
14. VMware, developer site, www.vmware.com
15. Kotenko, I., Doynikova, E.: Evaluation of Computer Network Security based on Attack Graphs and Security Event Processing. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), Vol.5, No.3, pp.14-29 (2014)
16. Fedorchenko, A., Kotenko, I., Chechulin, A.: Integrated repository of security information for network security evaluation. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), Vol. 6, No.2, pp.41-57 (2015)
17. TCP SYN Flooding and IP Spoofing Attacks. CA-1996-21, http://www.cert.org/historical/advisories/CA-1996-21.cfm