# SIMULATION OF THE COMBINED METHOD

Ilya Levin[1] and Victor Yakovlev[2]

[1]The Department of Information Security of Systems, State University of Telecommunication, St.Petersburg, Russia
lyowin@gmail.com

[2] The Department of Information Security of Systems, State University of Telecommunication, St.Petersburg, Russia
viyak@bk.ru

## ABSTRACT

*DDoS attacks have become one of the most dangerous issues in the Internet today. Because of these attacks, legitimate users can not access the resources they need. In [1] authors proposed a combined method for tracing and blocking the sources of DDoS-attacks. The essence of the method is that each router marks the network packet that passes through it using a random hash function from the set. At the receiving side this information is stored and used to filter unwanted traffic and traceback the source of distributed attack. This article describes the simulation and its results of the combined method.*

## KEYWORDS

*Traceback, DDoS-attacks, Attack path reconstruction*

## 1. INTRODUCTION

With the spread of network technology has increased the number of intruders attempting to prevent the normal operation of network applications and resources. One of the most widespread threats on the Internet today are DDoS-attacks (Distributed Denial of Service). DDoS-attack is directed at the violation or the complete cessation of legitimate users' access to networks, servers, services and other resources. To implement such an attack attackers create a bot-network (network of infected computers) and use them for attack by sending a huge number of packets from infected machines. Botnet can include hundreds or thousands of hosts spreaded around the world. According to Arbor Networks' Worldwide infrastructure security report the size of the largest single attack increased from 10Gbps in 2005 to 100Gbps in 2010 (1000 percent increase).

Because these attacks are performed using a wide range of addresses (including spoofed IP-addresses) it's very difficult to detect and block such attacks. Existing methods of protecting from DDoS-attacks do not have the high speed and automaticity of work.

A large number of DDoS-attack traceback systems were proposed [6-15]. Most of them are based on the principle of packet marking by routers [10-15]. Marks contain information about the part of the path that packet had passed. Rarely utilized fields of IP packet header are used for marks storing.

Marks can be made as mandatory by each router (deterministic packet mark) [12-14] and with a certain probability (probabilistic packet mark). A victim analyzing marks in received packets will be able to reconstruct the path a packet passed in the network and compute the attacking host.

The proposed method of tracing DDoS-attack sources combines several methods used on the routers (in particular, the method of mandatory packet marking using random hash functions) and the method used on the hosts.

The rest of this paper is organised as follows. In Section 2, description of the proposed method is presented. We introduce parameters of the combined method in Section 3. Section 4 demonstrates our simulation methodology. In Section 5, the simulation and results obtained using OMNet++ tool are discussed. Finally, we will conclude our paper in Section 6.

## 2. DESCRIPTION OF THE METHOD

Let's consider the way of packet marking. The mark is a hash value of the IP address of the router $R_i$ that processes a packet and is defined as $h_j^i = h_j(IP(R_i))$ where j is the number of used hash function. Marks are inserted into specially designated field in the IP header. The structure of the mark field is shown in Figure 1.

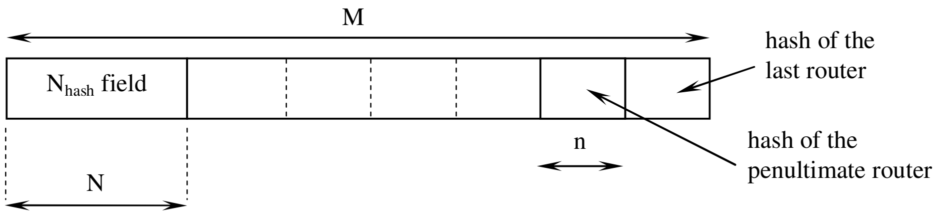

Figure 1. The structure of a mark field.

The total length of the field is M bits and it is divided into several parts. The first part $N_{hash}$ is for hash function number which was chosen by the first router that sent the packet. The length of this field is N bits. The second part is the field of marks which consists of a set of marks of length n bits which are inserted into the packet by routers. The maximum amount of marks in the packet is $S = \left\lfloor \frac{M-N}{n} \right\rfloor$, where $\lfloor \ \rfloor$ is the integer part of the fraction $\frac{M-N}{n}$ and it's called the depth of the traceable path.

Our task will be to optimize the parameters N, n and S for a fixed value of M. The value of M depends on the number of "free" bits in the IP header. The rarely utilized identification field (16 bits) and high-order bit of flags field (a total of 17 bits) are commonly used for IPv4. Identification field is used only in case of fragmentation of the package. The average fragmentation rate is not more than 0.25% of all packets [9]. In IPv6 for these purposes you can use the flow type field (20 bits) whose purpose is not strictly regulated by the standard. In [15] it is proposed to expand the field up to 25 bits.

Each router inserts its mark in the last n bits of the mark field. Marks made by previous routers are shifted to the left. The sequence of marks made by different routers $h_j^1$, $h_j^2$,..., $h_j^S$ by using the same hash function will be called the mark-vector. When the attacker attacks the victim it sends a packet, this packet arrives at the first router R1. It determines that the packet was sent from its autonomous system (AS) and initiates the process of packet marking. The router resets the field in the IP header, which is reserved for marks. It randomly selects number of hash function (j) which will be used for mark generating and computes the hash $h_j^1$ of its IP address using the selected hash function and writes it to the least significant bits of the mark field. Then it sends the packet to the next router. When the packet reaches the second border router R2, it determines that the packet is not from its AS and that the packet is already marked. The router reads the value of $N_{hash}$ field (which is equal to j) and computes the hash $h_j^2$ of its IP address. Then it writes the hash to the mark field of the IP packet shifting the previous hash left. Thereafter it sends the packet further. This process continues until the packet reaches the recipient (victim). At the end of the path the mark field will contain the hash codes of all the border routers through which the packet had passed.

If the value of S less than the number of border routers which the packet passes through, the left-most mark is deleted to give the opportunity to add new mark. In this case, surely the result of the operation of the method will contain an inaccuracy.

Victim stores all the necessary information from arrived packets in the specially designed databases. When the victim notices any abnormal activity (e.g., using intrusion detection system), it forms a request packet. This packet contains the following fields: field D is for measuring distance the request passed, field NH is the number of mark-vectors in the request, victim's IP address field, ID field is used to detect loops during the request transferring, marks table field contains mark-vectors and the numbers of hash functions which were used.

Victim sends this request packet to the nearest router RS. Router RS receives the request and reads the value of field D, increases this field by one and checks the appropriate column of the mark table with hash codes of border routers with which it is connected. Sends a request to only those routers for which there is a hash coincidence. If there is no match for any of the routers then the request packet transferring is terminated. When a request is sent simultaneously to two or more routers then their IP-address are stored in the option field to prevent the sending of the request to each other in the future. Other routers send a request in the same way to other routers increasing the value of D field by one.

When the request comes to the last border router it blocks the traffic sent to the host whose address is specified in the request or acts in accordance with the instructions contained in the option field of the request packet. Also it sends a response to the victim with its IP address, so the victim could send a request to remove the restriction on its address.

Request packet must be protected cryptographically to prevent it from spoofing. Consideration of this issue is beyond the scope of this article.

## 3. THE PROBABILITY OF FALSE DETECTION

The probability of false detection of the source of the attack Pfalse is one of the most important parameters of the combined method. Parameter Pfalse shows the probability of collision of marks from two different routers which as a result may lead to incorrect calculation of the path and blocking legitimate user traffic instead of traffic of attacker. The value of Pfalse depends on the length of the hash code generated by routers and the algorithm of generating mark. In [1] it was shown that this parameter for the case when the network is a binary tree can be calculated using the following expression

$$P_{false} \approx \left(\frac{1}{2^n}\right)^{2^N} \tag{1}$$

where n is the length of hash code.

It's very difficult to calculate Pfalse when routers have more than 2 connections. In this case combined method simulation is required.

## 4. SIMULATION

To test the performance and characteristics of the proposed method the program was written that allows to emulate the network and the method. The program should do:

– generate networks of different depths;
– delete or add links between routers to simulate different situations in the network;
– provide the possibility of changing the main parameters of the algorithm, such as: N, n;

– display the results of the attack tracing, the number of correctly detected, falsely detected and undetected sources of the attack;
– evaluate a large number of experiments with the networks of different topologies to obtain statistical results.

We will consider two types of network topologies:
– Near-binary tree network (network in which each node has an average of 2 children);
– Network generated based on the Euclidean graph.

In case of near-binary tree network, building of the network starts from the first node - the victim. Then each router is assigned randomly from one to three children. This process continues until the network reaches the depth that has been configured. Figure 2 shows an example of the process of generating a near-binary tree network.

We need networks of this type to check the correctness of the formulas for calculating the value of the probability $P_{false}$.

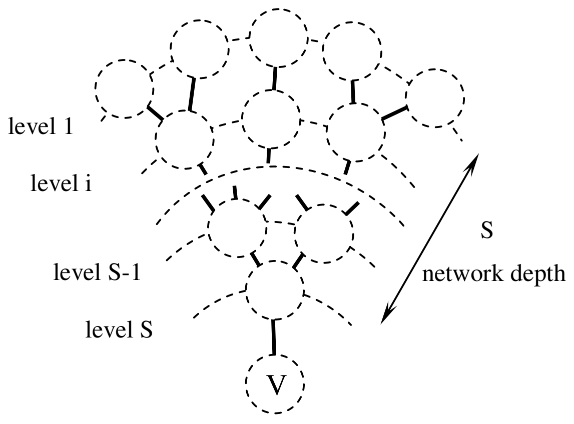

Figure 2. Example of the near-binary tree network generating process.

In case of network generated based on the Euclidean graph, routers are distributed on the surface randomly. Then calculated the average distance between all the routers. If the distance between two routers less than a specified threshold then these two routers are interconnected. By adjusting the threshold we can change the number of connections between nodes in the generated network. Thus we have areas in which the routers are connected. Then these areas are connected between each other. The result is the network which has areas of varying degrees of connectivity. This model is well suited for real-life network topology generating. Fig. 3 shows an example of the network obtained using the program.

The generated network consists of routers and lines connecting them. You can set any router as an attacker in the generated network (of course, we assume that the attacker is a host which is connected to this router). When attackers are selected the program emulates sending packets by each of them to the victim. Each router adds its mark to a virtual packet. When the packet reaches the victim the mark-vector is stored.

When the attack is done the sources can be traced. The program displays a summary table showing all detected sources. It indicates the number of real sources of the attack, the number of detected sources, the number of falsely detected and the number of undetected sources.

The program allows you to perform numbers of experiments to collect statistics. It automatically generates new network, performs an attack, traces the source of the attack and writes the results into the file. For each method configuration 10000 experiments were performed. Each experiment included 500 attackers.

The simulation results and the calculated characteristics are presented in Table 1. Obtained experimental results are comparable to the calculated values of the probability $P_{false}$.

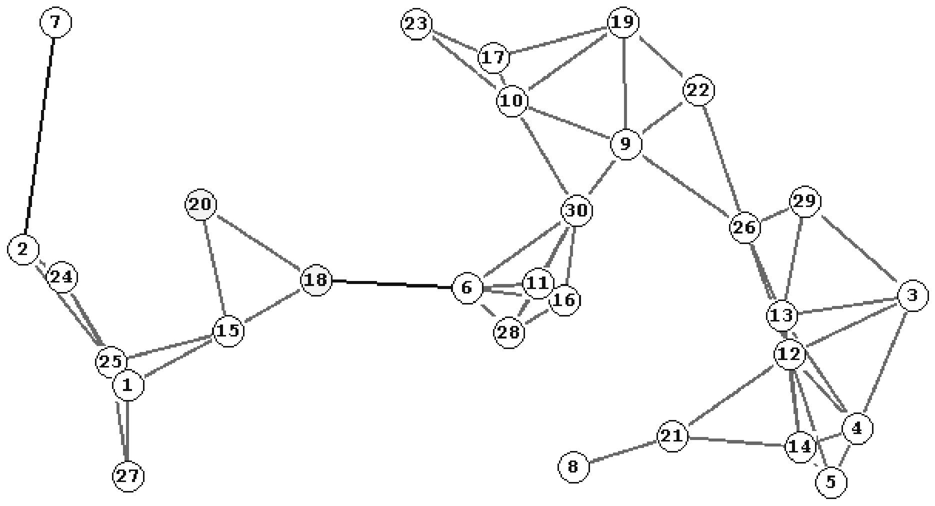

Figure 3. Example of the generated network based on Euclidean graph.

Table 1. Configuration options and method simulation results with M = 17 bits.

| № | Number of hash functions | n, bit | S | $P_{false}$ | | |
|---|---|---|---|---|---|---|
| | | | | Calculated | Simulation for tree networks | Simulation of networks constructed from the Euclidean graphs |
| 1 | 2 | 1 | 16 | 0,25 | 0,33 | 0,58 |
| 2 | 2 | 2 | 8 | 0,0625 | 0,07 | 0,081 |
| 3 | 4 | 1 | 15 | 0,0625 | 0,065 | 0,08 |
| 4 | 8 | 1 | 14 | $3{,}91*10^{-3}$ | $3*10^{-3}$ | $4{,}1*10^{-3}$ |
| 5 | 8 | 2 | 7 | $1{,}53*10^{-5}$ | $1{,}8*10^{-5}$ | $2{,}7*10^{-5}$ |
| 6 | 16 | 1 | 13 | $1{,}53*10^{-5}$ | $1{,}4*10^{-5}$ | $2*10^{-5}$ |
| 7 | 32 | 1 | 12 | $2{,}33*10^{-10}$ | - | - |
| 8 | 32 | 2 | 6 | $5{,}42*10^{-20}$ | - | - |
| 9 | 32 | 3 | 4 | $1{,}26*10^{-29}$ | - | - |
| 10 | 64 | 1 | 11 | $5{,}42*10^{-20}$ | - | - |
| 11 | 128 | 1 | 10 | $2{,}94*10^{-39}$ | - | - |
| 12 | 128 | 2 | 5 | $8{,}64*10^{-78}$ | - | - |

Because of small numbers of used hash functions options 1, 2 and 3 have a high probability of false detection. Therefore, they are not suitable. Options 8, 9 and 12 have a small probability of false detection but use a large number of hash functions. As a result the depth of the investigated network is small. Options 10 and 11 provide a low probability of false detection, greater depth of the network but because of the large number of hash functions may require additional time for processing.

The most suitable for use are options 6 and 7. They have a small probability of false detection and provide great depth of network investigation.

It's seen that theoretical results and experimental results converge well. Therefore we can assume that the theoretical results for which we were not able to do simulation due to small magnitude of probability are fair as well.

Despite this program can produce a large number of experiments it allows you to test the combined method with a relatively small amount of attackers (500 attackers). To perform a larger simulation, Omnet++ simulator was selected.

## 5. OMNET SIMULATION

OMNet++ is a discrete event simulator [17]. Events taking place inside the simple modules that are implemented directly by the user using C++ language. To describe the topology of the network NED (Network Descriptioin) language is used which is translated into C++. Because the network model is obtained entirely written in C++ and compiled into a binary executable file, it provides high-speed simulation even on large network models.

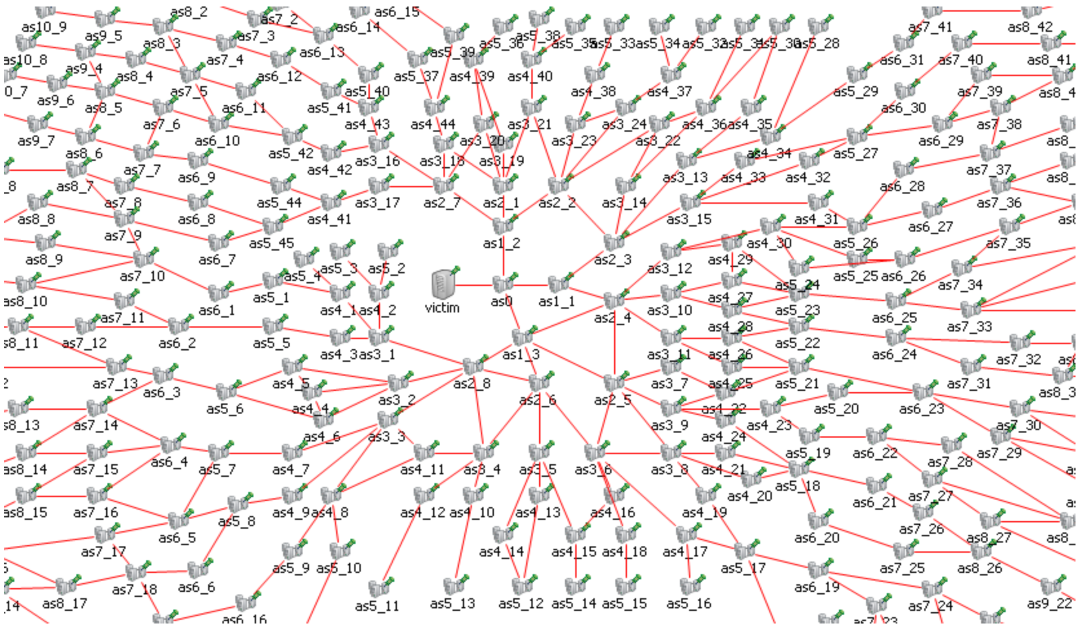

Figure 4. Example of the simulated network

Figure 4 shows an example of large network which consists of autonomous systems (AS). Each AS has a random internal structure (see fig. 5). Designation of autonomous systems looks like $as_{ij}$. This means that the AS is at a distance i from the Victim and this is the $j^{th}$ AS at this distance.

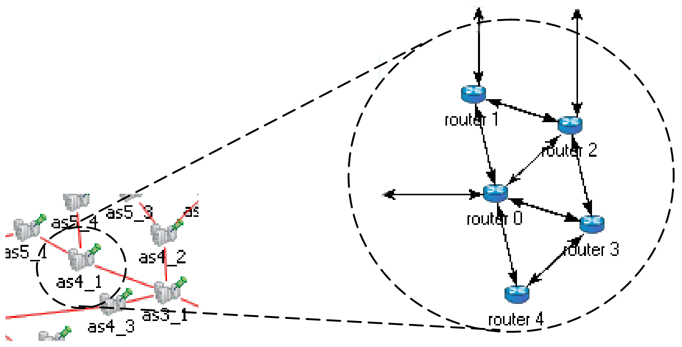

Figure 5. Internal structure of the AS

The following parameters were chosen during simulations:

– number of routers – 1500;
– number of legitimate users – 1000;
– number of attackers – 25000.

Table 2 shows the simulation results of the combined method using Omnet++. It is seen that for large-scale networks in which the number of connections between nodes is random and often more than two, the probability of false detection is much higher than was calculated for configurations for which the probability of collisions of hash functions is high (options 1-4).

Table 2. Configurations and method simulation results using Omnet++ (M = 17 bits)

| **№** | **Number of hash functions** | **n, bit** | **S** | **Pfalse** | |
|---|---|---|---|---|---|
| | | | | **Calculated** | **Simulation** |
| 1 | 2 | 1 | 16 | 0,25 | 0.81 |
| 2 | 2 | 2 | 8 | 0,0625 | 0.12 |
| 3 | 4 | 1 | 15 | 0,0625 | 0.13 |
| 4 | 8 | 1 | 14 | $3,91*10^{-3}$ | 0.02 |
| 5 | 8 | 2 | 7 | $1,53*10^{-5}$ | $2.7*10^{-3}$ |
| 6 | 16 | 1 | 13 | $1,53*10^{-5}$ | 0 |
| 7 | 32 | 1 | 12 | $2,33*10^{-10}$ | - |

Configurations with a higher number of used hash functions have shown good coincidence between the results obtained by calculation and simulation (options 5-6). For configurations of the method for which the calculated probability of false detection $P_{false}$ was less than 10-9 the simulation was not performed.

Also, using this program dynamic characteristics of the combined method were investigated.

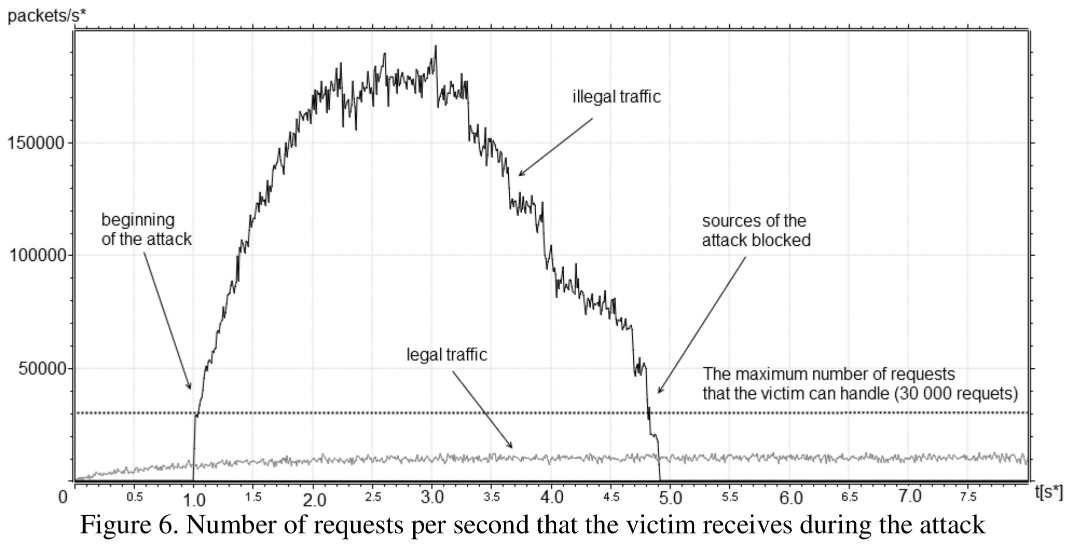

Figure 6. Number of requests per second that the victim receives during the attack

For each method configuration 20 experiments were performed. Figure 6 shows that traffic from the attackers (black line) much more than traffic from legitimate users (gray line). Figure 6 and 7 were obtained during testing option 5 from Table 2. Figure 7 shows that during the attack victim can not handle all the requests coming from legitimate users, so some of them are dropped. Once the sources of the attack tracked and blocked, the victim able to handle all requests from legitimate users again.

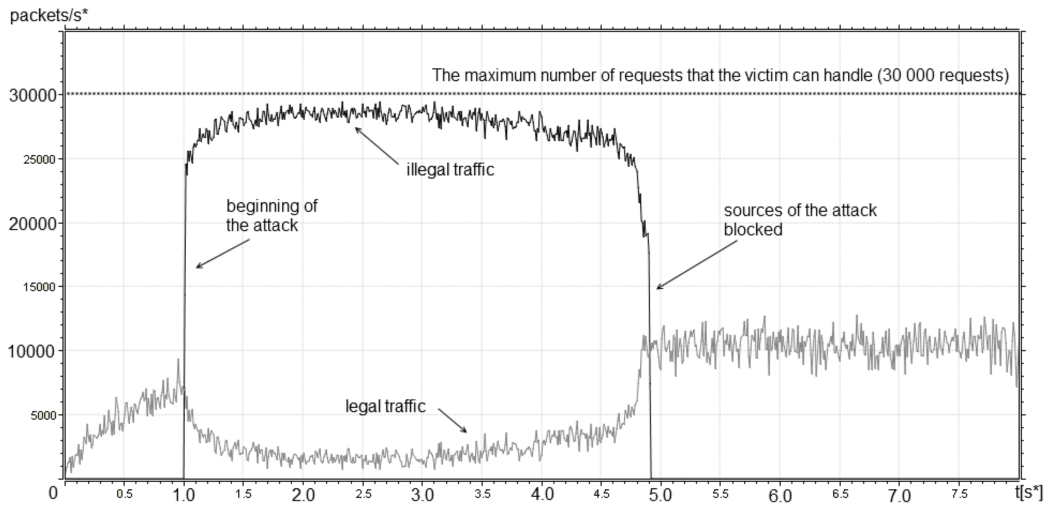

Figure 7. Number of requests per second that the victim could process during the attack

## 6. CONCLUSIONS

The results of the simulations revealed the following:

1. The combined method allows to trace and to block the sources of DDoS-attack in large-scale networks. During testing, there was no case where the real source of the attack was not detected. The exceptions were cases where the source of the attack was at a distance farther than the combined method allows you to track. This should be considered when choosing a method's parameters.
2. Were found the most appropriate configuration parameters for the method to ensure the desired range of attack detection and satisfy the requirements of probability of false detection.
3. Analytical conclusions were confirmed for the probability of false detection of attack sources on networks. Also noted that for real networks in which the number of connections between nodes is random, the probability of false detection is slightly higher and depends on the configuration of the network.

## REFERENCES

[1] V.Yakovlev, I. Levin, (2010) Combined Method of Tracing DDOS-attack Sources, Information security problems. Computer Systems, pages 48-61.

[2] V.Yakovlev, I. Levin, (2010) Simulation and performance study of the combined method for tracing DDoS-attacks, Proceedings of the Ninth International Symposium on Intelligent Systems, pages 584-588.

[3] Lin, C.-H., Liu, J.-C., and Kuo, C.-T., (2010) Priority Queue-based Scheme to Maintain Quality of Service for Normal Users Suffering from Large DDoS Attacks, International Journal of Future Generation Communication and Networking Vol. 3, No. 2.

[4] Y.Bhavani, P.N. Reddy, (2010) An efficient IP traceback through packet marking algorithm, International Journal of Network Security & Its Applications (IJNSA), Vol.2, No.3.

[5] M.-C. Lee, Y.-J. He, Z. Chen, (2009) Towards Improving an Algebraic Marking Scheme for Tracing DDoS Attacks, International Journal of Network Security, Vol.9, No.3, pp.204–213.

[6] P. Ferguson, D. Senie, (1998) Network Ingress Filtering: Defeating Denial of Service Attacks which employs IP Source Address Spoofing, RFC 2267.

[7] H. Lee, K. Park, (2001) On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack. In Proceedings IEEE Infocomm.

[8] S. M. Bellovin, (2000) ICMP traceback messages. In Work in Progress, Internet Draft draft-bellovin-itrace-00.txt.

[9] S. Savage, D. Wetherall, A. Karlin, T. Anderson, (2000) Practical Network Support for IP Traceback. In ACM SIGCOMM.

[10] D. Song, A. Perrig, (2001) Advanced and authenticated marking schemes for IP traceback. In Proceedings IEEE Infocom.

[11] T. Peng, C. Leckie, (2002) Adjusted Probabilistic Packet Marking for IP Traceback.

[12] A. Belenky, N. Ansari, (2003) IP Traceback with Deterministic Packet Marking, IEEE COMMUNICATIONS LETTERS, VOL. 7, NO. 4.

[13] A. Yaar, A. Perrig, D. Song, (2003) Pi: A Path Identification mechanism to defend against DDoS attacks. In Proceedings of IEEE Symposium on Security and Privacy.

[14] A. Yaar, A. Perrig, D. Song, (2004) StackPi: New packet marking and filtering mechanisms for DDoS and IP spoofing defence, In IEEE Symposium on Security and Privacy.

[15] S. Karthik et al., (2008) Multi Directional Geographical Traceback with n Directions Generalization, Journal of Computer Science 4 (8): 646-651.

[16] I. J. Carter and M. N. Wegman, (1979) Universal classes of hash functions. J. of Comput. Syst. Sci., vol.18, pp.143-145.

[17] OMNeT++ homepage. http://www.omnetpp.org/

[18] The Cooperative Association for Internet Data Analysis, www.caida.org

[19] Y. Wu, H. Tseng, W. Yang, (2011) DDoS detection and traceback with decision tree and grey relational analysis, Int. J. Ad Hoc and Ubiquitous Computing, Vol. 7, No. 2.

[20] N. Arumugam, C. Venkatesh, (2011) A Trivial Scheme for Detecting and Preventing Fake IP Access of Network Server Using IPHP Filter, European Journal of Scientific Research ISSN 1450-216X Vol.53 No.2, pp.258-268

[21] P. Wang, H. Lin, T. Wang, P. Kuo, (2011) A New Approach for Solving the IP Traceback Problem for Web Security, Advances on Information Sciences and Service Sciences. Volume 3, Number 2, March 2011.

[22] K. Garg , R. Chawla, (2011) Detection of DDoS attacks using data mining, International Journal of Computing and Business Research (IJCBR), Volume 2 Issue 1 2011.

[23] P.Sundari, Dr.K.Thangadurai, (2010) An Empirical Study on Data Mining Applications, Global Journal of Computer Science and Technology, Vol. 10, Issue 5, pp23-27 July 2010.

[24] Shui Yu, W. Zhou, R. Doss, W. Jia, (2011) Traceback of DDoS Attacks Using Entropy Variations, IEEE Transactions on parallel and distributed systems, Vol. 22, No. 3, March 2011.

[25] M. Sung et al., (2008) Large-Scale IP Traceback in High-Speed Internet: Practical Techniques and Information-Theoretic Foundation, IEEE/ACM Trans. Networking, vol. 16, no. 6, pp. 1253-1266, Dec. 2008.

[26] M.T. Goodrich, (2008) Probabilistic Packet Marking for Large-Scale IP Traceback, IEEE/ACM Trans. Networking, vol. 16, no. 1, pp. 15-24, Feb. 2008.

**Authors**

Viktor Yakovlev received the MSc degree in Electrical Engineering from the Military Communication College, Kiev, Soviet Union, in 1972, the PhD degree and the Doctor in Philosophy degree from the Military Communication Academy in 1981 and 1994 respectively. He has been professor at the Military Communication Academy, and at the State University of Telecommunications, both in St. Petersburg, Russia. His main research interests are in coding theory, cryptography and information security

Ilya Levin received the MSc degree in Electrical Engineering from the State University of Telecommunications, Saint-Petersburg, Russia, in 2007. Currently he is writing a dissertation at the same university on theme of protection from DDoS-attacks. His main research interests are in information security, DDoS-attacks protection techniques.