# Application of SDN Technologies to Protect Against Network Intrusions

A.S. Salimov
Crimean Federal University
Simferopol, Russia
E-mail: ars.salimov@gmail.com

N.M. Dolgopolov
Samara University
Samara, Russia
E-mail: kamikadzni@mail.ru

A.M. Sukhov
Samara University
Samara, Russia
E-mail: sukhov@ssau.ru

E.S. Sagatov
Samara University
Samara, Russia
E-mail: sagatov@ya.ru

*Abstract* — **This paper discusses the construction of an infrastructure for protecting against network intrusions based on SDN technologies. Protection technologies were developed earlier using rank distributions, and they allow for the finding of threshold values for the basic network variables, compile lists of regular users of the resource, and prepare databases of IP addresses from which intrusions are made in advance. It is proposed that all these methods can be improved by applying SDN technology, and an SDN infrastructure has been deployed for experimental testing of software modules being developed.**

***Keywords — protection based on SDN technologies, network monitoring systems for intrusion detection, rank distribution to detect attacks***

## I. Introduction

A major danger on the Internet is the actions of various kinds of intruders. They use the global network not only as a tool to commit unlawful actions but also to hamper the normal operation of core network applications. Types of network attacks are constantly being improved upon, and often the actions of intruders are well coordinated [1].

To counteract coordinated attacks, an equally coordinated system of intrusion protection is required. The whole issue revolves around how coordination takes place. More recently, network operating systems that were installed on network routers, such as Cisco IOS, could not offer enough tools to coordinate efforts to prevent network intrusions. Therefore, confrontation with modern DDoS attacks, aimed at overflowing external channels for hosting services, was difficult.

The emergence of SDN technologies [2], in which the control plane and the data plane are separated, makes it easy to coordinate protection actions for a whole group of network devices. SDN technologies allow for a single policy for traffic management and the formulation of uniform rules, as well as the introduction of unified permissive and prohibitive lists of IP addresses.

Some types of attacks require you to impose restrictions on traffic on routers that are located two levels higher than the attacked hosting. Such incidents include the most dangerous type of DDoS attacks and attacks on the overflow of external communication channels. At the same time, several routers can be managed by third-party providers that conduct an independent security policy, but even here SDN technologies can come to the rescue.

As an example of such interactions, one can cite the preparation of the Defense Information Systems Agency (DISA) of the US Department of Defense (DoD) to repel attacks with a power of several terabits per second [3]. When creating a protective infrastructure against terabit attacks, SDN/NDV technologies are supposed to be used.

This article is about attempts to create a security infrastructure based on SDN technologies in Russia. The field of interest of our group included the development of network security systems. Our efforts made it possible to achieve a number of results in several areas, such as:

- Detecting attacking IP addresses by exceeding the threshold values for the main network variables and countering them.
- Drawing up permissive lists of regular Internet service users.
- Using a honeypot server to compile databases of attacking addresses and model attacks with the allocation of a ranked by popularity list of threats.

All these studies were performed initially without the use of SDN technologies, so data processing was quite a difficult exercise and was carried out manually on the data collected over several months. This article will discuss the issues of how to modernize the hardware schemes of the mentioned experiments and the protection infrastructure,

**978-1-5386-9456-5/18/$31.00 ©2018 IEEE**

including elements of SDN technologies. Our discussion also includes the development of appropriate SDN modules.

## II. Network monitoring systems for intrusion detection

In this section, we will describe in detail the schemes of experiments that can detect sources of network intrusions.

Let's start with methods that allow for identifying the beginning of DDoS (Distributed Denial of Service) attacks [4, 5], as well as the network addresses from which this attack is conducted. This method assumes that for several network variables there is a limit value that is significantly exceeded during the attack. This assumption is based on the fact that during the attack there is a denial of service due to the artificially created load on the network resources, although in the normal state the attacked service handles the flow of requests calmly. Therefore, during an attack, the values of network variables must exceed a certain barrier. Through a set of variables that exceed this barrier, it can judge a particular type of attack.

In this section, it is necessary to discuss the rules for calculating threshold values. Also, it is necessary to establish methods of measuring them in order to quickly determine the moment of the beginning of the attack. The idea of determining thresholds for the detection of anomalous network states was expressed long ago, since the initial appearance of DoS attacks [6]. For the calculation of threshold values, rank distributions are used when the values of the network variable measured at the same time are arranged in descending order.

In 1994, Steve Glassman [7] first described the process of Internet traffic caching using rank distribution. In the future, the field of application of rank distributions for the analysis of network processes has expanded. At present, there are several good reviews [8, 9] on the use of rank distributions for the description of network processes.

As a rule, Internet processes at a fixed time are described by a Zipf-like distribution which states that

$$p_i = \frac{p_1}{i^{\alpha}} \tag{1}$$

here $p_1$ is the largest value of the investigated variable, $i$ is the sequence number in the ranked list (list in descending order), and $\alpha$ is the exponent characterizing the rate of decrease.

To detect the moment of the beginning of the attack, the value $k$ should be used

$$k = \frac{p_1}{p_{tr}}. \tag{2}$$

Now the question is how to determine the threshold value $p_{tr}$, which stands in the denominator of the fraction from equation (2).

For this it is necessary to construct the dependence of the largest value of the investigated network variable on time, $p_1(t)$. First, this largest value is calculated at the current time, and then its time dependence is constructed. To calculate the threshold value, we need to collect and process statistics for a significant period. This period should be at least a week to account for all temporary fluctuations. Then the maximum value of the function $p_1(t)$ on the investigated interval should be taken as the desired threshold value. The threshold value $p_{tr}$ should not be exceeded during normal network operation ($p_1(t) \le p_{tr}$). This value is used to calculate the coefficient $k$ in equation (2).

The next question is to determine the set of network variables for which threshold values are to be calculated. The choice of network variables depends on the type of DDoS attack. If the attack is aimed at breaking an Internet service (for example, failure in the functioning of a web server), then we should analyze the number of requests to the attacked resource. If the attack targets the overflow of incoming channels, then it is required to collect data about all types of incoming traffic (TCP, UDP, ICMP), as well as information on the number of active streams.

Since the attack type is not known in advance, threshold values must be calculated for a significant number of variables. Such a set of variables should include:

- the total number of active flows on the border (BGP) router;
- the number of active flows that generate a single external IP address;
- incoming traffic that generates a single external IP address, separately for each type of traffic (TCP, UDP, ICMP);
- the number of requests generated by a single external IP address, separately for each type of service (HTTP, FTP, mail, proxy, ssh, samba, MySQL, etc.)

After threshold values $p_{tr}$ for the most important network variables are found, it is necessary to regularly calculate the corresponding values of the coefficients given by equation (2). If the value of this coefficient is much greater than one, we should speak about the anomalous state of the network. The fact that all network variables are calculated for traffic that generates a single IP address allows us to determine the sources of attack.

As an illustration, the Netflow data from the external routing server of the Samara State Aerospace University is used. The obtained results are shown graphically in Fig. 1.

A typical graph of the distribution of the number of active flows generated by a single IP address is shown in Fig. 1. The number of active flows and the ordinal number of the address in the ranked list are plotted along the axes in a logarithmic scale. The points on the graph lie

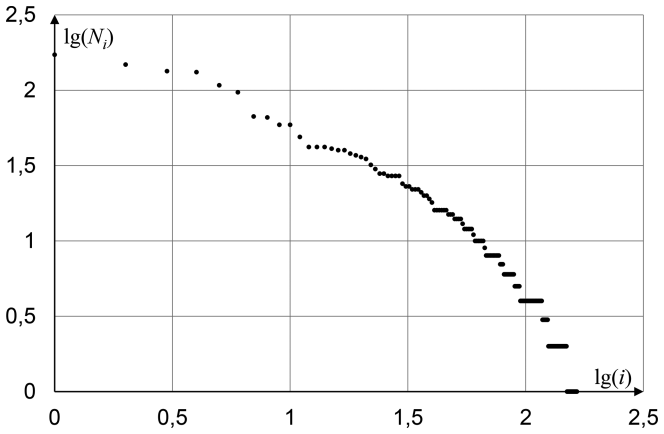


**Fig. 1.** The number of active flows per IP-address in descending order

on a straight line, which indicates that the given distribution obeys Zipf's law from Equation (1).

To find the threshold for the number of active flows generated by a single IP address, the weekly traffic was analyzed and the corresponding graph is shown in Fig. 2.

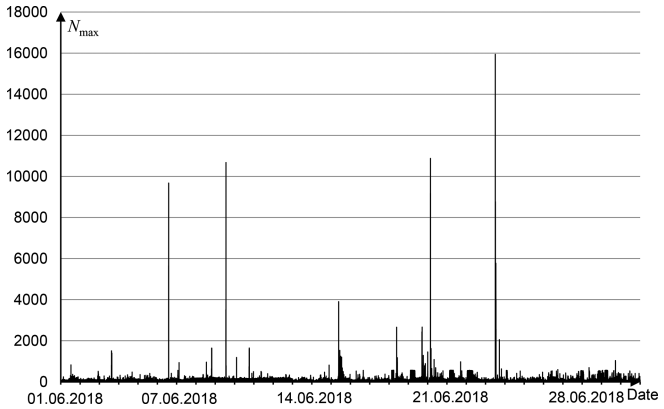


**Fig. 2.** Dynamics of changes in the number of active flows

The graph in Fig. 2 shows that the maximum number of flows from one IP address during normal network operation does not exceed 600 in 5 minutes for the server under investigation.

Thus, based on Zipf's law and the flow data from the routers, we formulated and illustrated the rule for determining the clipping threshold for the number of flows for suspicious IP addresses. The clipping level will be the upper limit among the maximum number of flows generated by a single IP address.

When repelling a DDoS attack, we can use different defense strategies, such as searching for classification characteristics of IP-addresses from which an attack occurs and restricting access to the protected service [10]. However, at the beginning of the attack, it takes some time to recognize the characteristics, formulate qualification characteristics for unwanted queries, and identify the attacking IP addresses. The duration of these operations can reach several hours [11], during which the service may be unavailable. At this time, we can only process requests from a regular audience, that is, a custom kernel. Knowledge of the statistical characteristics of the behavior of new users will facilitate the search for types of attacking requests and the formation of lists of attacking IP-addresses for subsequent blocking.

To identify the user's kernel of the Internet service [12], we can use any access logs to which visitors' IP addresses are entered. In this paper, we investigated the web server of the popular Internet portal, so the log files of the Nginx web server running on the Debian GNU/Linux operating system are used as the source of the IP addresses. IP-addresses of visitors are entered in the database and are further considered to belong to the user's kernel. After the kernel is formed, users whose IP addresses are not included in the database are considered new.

Figure 3 shows the time dependence of the share of new IP addresses, η, in the daily audience. The proportion η is the ratio of the number of IP addresses of new visitors to their total number for each day.

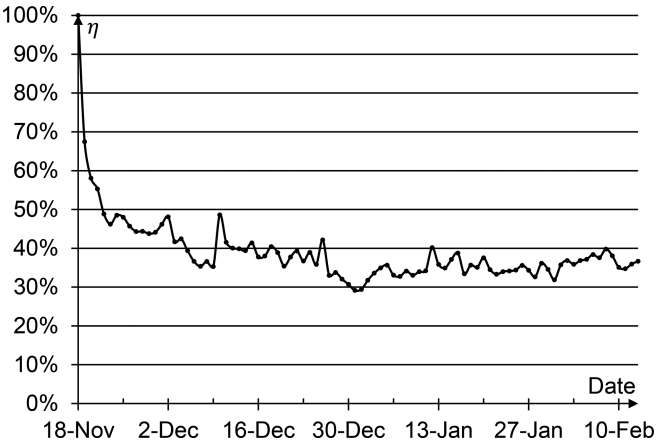


**Fig. 3.** The graph of the percentage of new IP addresses

It can be seen from the graph that, in four to six weeks from the beginning of the statistics collection, the percentage of new IP addresses from which site visits occur is stabilized in the range of 30 to 40%.

In order to understand the structure of the audience, it is necessary to build ranked lists of user IP addresses [8]. The ordering frequency is chosen to be the frequency of visits $R$ from each address, and the user IP addresses are arranged in decreasing order of $R$. As the rank of $n$, the sequence number of the IP address in the descending list will be used, and then the function $R(n)$ will be the desired rank distribution.

The Nginx web server logs allow us to obtain the Zipf-like distribution shown in Fig. 4.

The study of the service audience, conducted above, allows for forming a list of regular visitors. The base of such a list is the audience for some significant period (from two weeks to several months). First of all, from this list it should delete all random addresses. These are the addresses from which they visited the site only for one day.

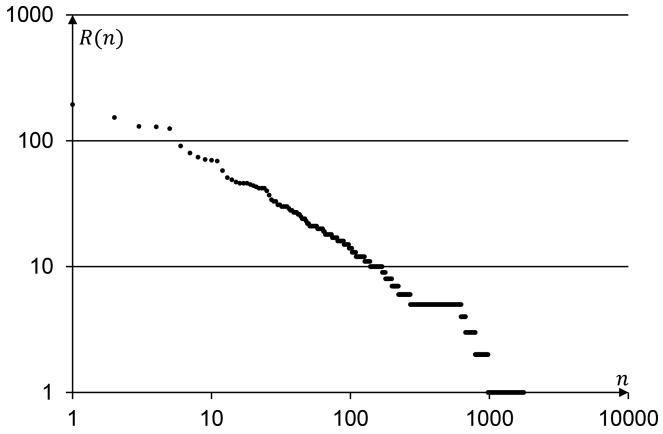


**Fig. 4.** Rank distribution of site visits

This list will not be complete, since some of the IP addresses are selected from blocks of dynamic Internet provider addresses. To develop the list of IP addresses, a special script was developed. It should be noted that the developed script for expanding the user database must be run before the previously described script for removing suspicious IP-addresses. The addresses of visitors from subnets belonging to the user's kernel are not considered accidental and are not deleted during cleaning.

The application of the developed script to supplement the database allowed us to increase the number of addresses included in it by 44%. After the core expansion, the share of new addresses was in the range of 25 to 35%.

To perform the prohibition list, a slightly different approach was used based on the concept of a honeypot. A honeypot is a server with a real IP address on which software is installed that implements a number of popular Internet services (Table 1). At the same time, the network data of these services and server addresses are not published anywhere, so that ordinary real users do not have the opportunity to learn about them. Thus, they can only be found via robotic search.

The task of the honeypot is to attract criminals, record their actions and then analyze them. Therefore, requests to such a server, especially multiple ones, can be classified as abnormal network activity.

The idea of a honeypot server was first conceived in 1990 [13], while the mass use of such technologies began in the early 2000s [14]. During this time, as the technology progressed there appeared specially developed software for these purposes, both open [15] and commercial. In our work, we will use software with an open license for all components of the honeypot server.

In order to improve the accuracy of predictions, it is necessary to install several such servers in different parts of the world. Further data from these servers will be synchronized. We selected three points within Russia: in Samara, Rostov-on-Don and Crimea, and also one point in the USA. The last point was chosen to compare the situation in Russia and the United States.

In addition to coverage, our research was different and conducted over a long enough time to collect statistics, which was more than a year. In the analysis of regularities, methods based on rank distributions were used.

The choice of applications installed on the honeypot server-was determined by their popularity with users, as well as the ability to organize the collection of call statistics in the simplest way.

All honeypot servers had the operating system GNU Debian/Linux installed. A list of protocols, services, associated software, types of attacks and registration files with their location are shown in Table 1.

When configuring network protocols and services, standard ports were used.

It should be noted that the analysis of the log files listed in Table 1 is not enough to structure the threats, since these files contain only the response of the system. In order to record the request itself, we need to install a

**Table 1.** The main parameters of the honeypot server

| № | Network protocol or service | Installed software | Possible types of attack | Path to the data file |
|---|---|---|---|---|
| 1 | VoIP, SIP | Asterisk | Selecting a password, incoming call to search for an existing number | /var/log/asterisk/messages |
| 2 | HTTP, Web service | Apache, Nginx | Trying to find the admin interface of phpMyAdmin, WordPress, Joomla; other queries with vulnerability search | /var/log/nginx/* |
| 3 | POP3, IMAP, e-mail | Dovecot, Exim | Choose a password | /var/log/mail.log |
| 4 | MySQL, database management system | MySQL | Choose a password | /var/log/mysql/* |
| 5 | SMB, universal access service to network resources | Samba | Choose a password | /var/log/samba/* |
| 6 | Web-proxy, proxy server with reservation option | Squid | Choose a password | /var/log/squid3/access.log |
| 7 | SSH, secure remote management | OpenSSH | Choose a password | /var/log/auth.log |
| 8 | FTP, File Transfer Protocol | vsftpd | Choose a password | /var/log/vsftpd.log |
| 9 | DNS, Domain Name System | Bind9 | DNS Vulnerabilities | /var/log/named.log |
| 10 | Firewall | iptables | Port scanning | /var/log/iptables |

program to record the incoming traffic on the network interface of the honeypot server. In the described case, the program WireShark was installed, which permitted analyzing incoming requests and classifying them according to the types of vulnerabilities. Subsequently, this information will be used to compile an audit plan for the security of information systems.

As a result of the analysis of log files, we have compiled a list of suspicious IP-addresses (black list). The criteria for entering this list include the following two assumptions:

1) A simultaneous hit of the IP-address in the log files on 2 or more honeypot servers.

2) Receiving at least 3 requests from each suspicious IP-address.

## III. Protection against network attacks

Any network attack is important to detect in a timely manner in order to take measures to neutralize it as quickly as possible. The network status data is received from the network devices periodically, depending on the settings set. At the same time, a balance is needed between the frequency of statistic collection and the time needed to process it. Therefore, the frequency of the data request is one time per minute or five minutes.

It should be noted that NetFlow-statistics contain information about already completed flows. Because the flow is considered active for a certain time after its completion, such completed flows also need to be considered active. A popular Internet portal has been moved here. Schematic diagram of the network infrastructure is shown in Fig. 5.

This infrastructure consisted of the following elements:

- A NetFlow sensor installed on the border router.
- A special script based on NetFlow, designed to isolate attacking IP addresses.
- A special script that determines the beginning of the attack on a sharp increase in the number of active flows.
- An additional Cisco 2811 router in front of the server being attacked, with stop-lists installed on it.
- A Switch 3Com 4500 for duplicating network traffic and its subsequent saving.
- A specially formed list of regular visitors, which was activated when the attack began to limit visitors to the site.

The allocation of the device to duplicate traffic was due to the fact that it is necessary to collect all traffic during the attack for further analysis. When entering commercial operation, access lists should be downloaded from an Internet provider of a higher level.

Before the tests in the conditions of a real attack, a complex laboratory test was conducted involving 10 attacking computers. They were located both in the network of the enterprise and outside it.

The attack on the number of requests to the web server was performed using an Apache HTTP server benchmarking tool [16], a Low Orbit Ion Cannon [17], and BoNeSi [18]. The UDP flood attack was carried out with the help of the Low Orbit Ion Cannon and BoNeSi.

None of the attacks disrupted the hardware, and the web server responded to user requests. It should be noted that the tests were carried out on a commercial system that was not a laboratory bench. Therefore, it was impos-

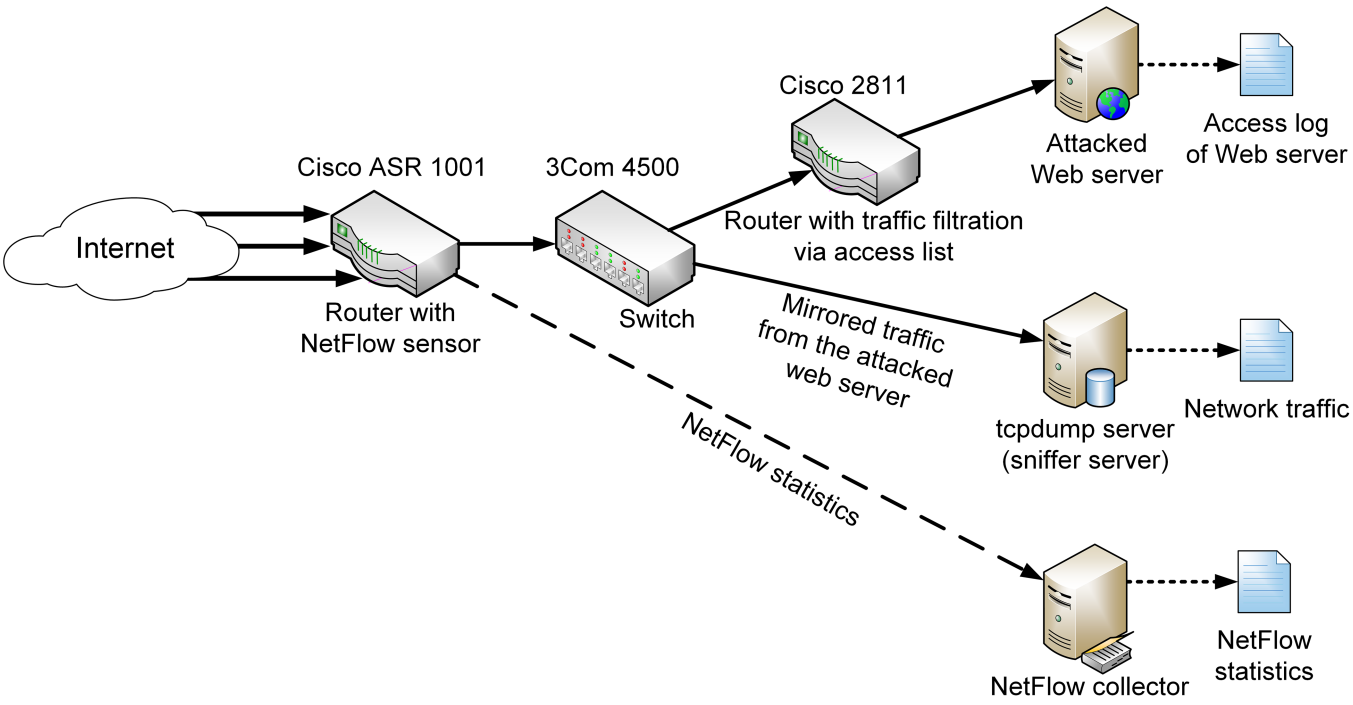


**Fig. 5.** Network infrastructure for research

sible to fully emulate a large botnet with the real participation of just a few attacking computers.

Lab tests cannot replace the experience gained from a real attack, so the authors anonymously requested a combined DDoS attack on the web server described above. Preliminary statistics of the use of this server were collected within five months.

The real experience of repelling the DDoS attack radically changed the authors' opinions about the type and features of the attack. Before the attack, it was planned to remotely monitor the equipment, but the first minutes of the DDoS attack showed that it was impossible to do this via the attacked communication channel. The beginning of the DDoS attack was accompanied by a sharp increase (more than a hundred times) in the number of active flows, which was recorded in time by scripts of observation. Then, during the first minutes of the DDoS attack, external communication channels overflowed, and the web server became unavailable from the external network. All other services and servers located in this local network also became inaccessible, and remote management was lost despite three external communication channels. Therefore, we had to switch to management from the internal network. Overflow of one of the external channels is demonstrated in the graph shown in Fig. 6.

The solid line indicates the maximum available bandwidth of the channel to the service provider. During the entire attack, it was significantly exceeded.

The failure of communication channels occurred due to overflow with incoming UDP traffic (DDoS-attack type "UDP flood"), and the number of servers generating this traffic was relatively small (only reaching around 200). About half of these servers practically did not change the source and destination ports, while the other half did it regularly.

During the attack, two types of UDP packets were used, the first being the minimum-length. These packets contain one character which is repeated in all packets. The second type is UDP-packets of maximum length. These packages are filled with random data, and all are marked as fragments for their subsequent integration by the server into one large package.

A small number of attacking servers were compensated for by the total speed of UDP streams. From some addresses, this speed reached 60 Mbps. This speed could be increased, but our provider imposed restrictions on the external channel. It was found that most of the attacking servers were located in the US, although correspondence with the botnet management was conducted in Russian. According to the management of the botnet, its attacking power was only used by 2%. In this case, only the web hosting, with its external channels of the order of 1 Gbps, is affected. The channels of our external provider with a total capacity of at least 100 Gbps are not affected.

Unfortunately, the hosting did not have an agreement to limit incoming traffic with providers, and the simplest ban of UDP traffic to the attacked server would immediately solve most of the problems.

TCP requests (DDoS-attack type "TCP flood") also participated in the DDoS-attack. The number of attacking servers was about 1500. During the attack, TCP requests of two types were used. The first was file requests from the web server, simulating user actions. The second type was represented by a lot of SYN/ACK-packages of the minimum size; apparently, it is a "TCP SYN flood" DDoS-attack. But these attacks did not inflict significant damage in view of channel overflow and activation of query restriction algorithms on the web server.

Analysis of the attack at the flow level showed that the beginning was accompanied by a sharp increase in the number of active flows in the external channel of web hosting. The number of completed flows, as mentioned above, has increased by more than two orders. This growth was immediately fixed by the monitoring system. Individual attacking IP addresses could easily be determined by the number of generated flows, both active and completed. Figure 7 shows the ranked list of addresses by the number of generated flows. The top graph describes the moment of attack, and the lower graph shows a typical distribution in the absence of an attack.

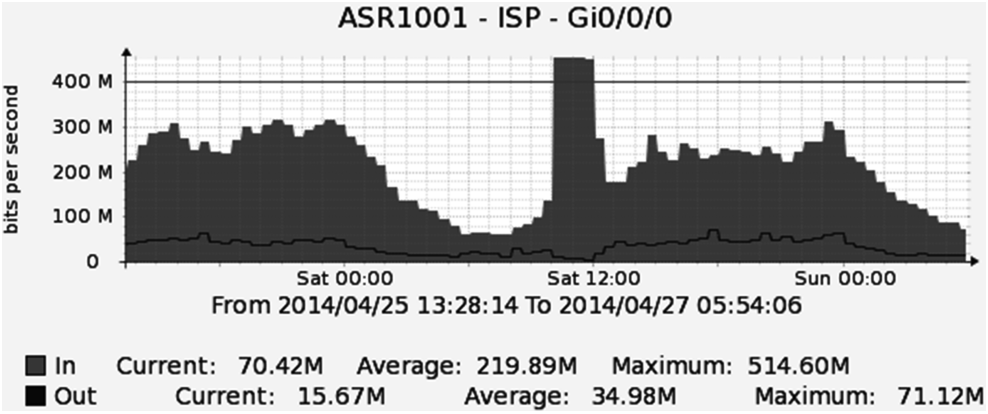


**Fig. 6.** The schedule for loading an external channel during a DDoS attack

Comparison of the graphs presented in Fig. 7 allows us to formulate a criterion for determining whether an IP address belongs to a botnet. All addresses located higher than the most popular server in the normal network state and not belonging to the user's kernel should be assigned to the botnet [19]. For full detection of attacking addresses, it is necessary to build ranked distributions for incoming UDP, ICMP and TCP traffic generated from a single IP address at the time of the attack. The cut-off threshold must be defined for each service and type of traffic in advance, and periodically (at least every six months) these values must be recalculated.

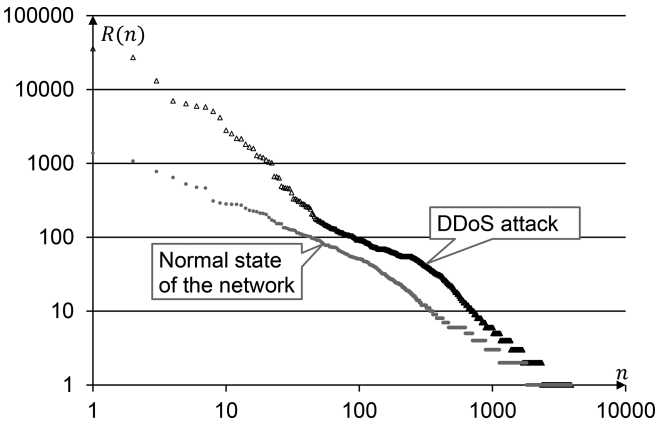


**Fig. 7.** The number of flows during the attack and in the normal state

The use of two independent criteria for the number of flows and the amount of incoming traffic (UDP, ICMP or TCP) allows us to quickly (within 5 minutes) make a list of attacking addresses for blocking on filtering equipment.

## IV. Schematic diagram of the SDN security infrastructure

In this section, the basic diagrams of software and hardware complexes based on SDN technologies will be presented. In the previous sections, we summarized our experience in identifying network intrusions and organizing security. Unfortunately, it turned out that the standard protection schemes do not work for the most terrible type of attacks on the overflow of external communication channels. To protect against this type of attack, we need to limit traffic on routers located one or two levels above the protected resource. In other words, these restrictions should be introduced at the provider's provider.

That is, the zone controller must collect and analyze information from the local switch. Based on this information, the initial data on the protected network should be collected and the main classification characteristics of network intrusions should be identified on their basis. These classification characteristics, in the form of threshold values for the main network variables and lists of persistent users of the service or databases of attacking addresses for a particular service must be transferred to higher routers. As a rule, these routers enter the network of the provider and are managed by its SDN controller, as shown in Fig. 8.

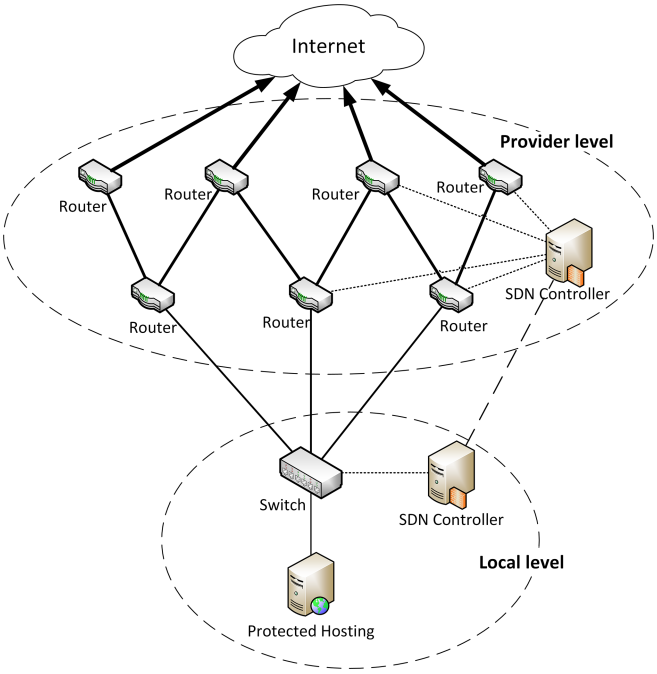


**Fig. 8.** Schematic diagram of the SDN security infrastructure

The main problem with this scheme is the organization of the interaction of SDN controllers. For security reasons, the provider cannot allow its network devices to receive control information from a third-party controller. To effectively protect against attacks, we must set restrictions on routers located two levels higher than the protected server. At the same time, among the higher-level network devices are routers of third-party providers. Therefore, it is necessary to transfer the results of the processed data between the controllers.

We had a problem in selecting equipment that supported the SDN protocols. At first we tried to buy the switch at the lowest price and settled on the Huawei S5720–12TP-LI-AC model. This model was offered by Huawei representatives after long negotiations. Together with the equipment, a certificate of compliance from the Open Flow Foundation from December 2016 on the support of OpenFlow 1.3 specifications was presented. Unfortunately, this switch did not support this technology, and our numerous calls to the support service did not help.

Therefore, the question arose about the urgent purchase of equipment for testing. Therefore, as an SDN switch, we purchased a Zodiac FX board, and as the SDN controller we chose Floodlight. However, this was a temporary solution; a few months later we were able to purchase the HP Aruba 2930F 4SPF switch. All experiments with writing modules were carried out on this equipment.

## V. Development and testing of SDN modules

In order to translate our protective mechanisms into the SDN area, we needed to develop the appropriate modules. Since this activity involves a large amount of work, we identified the most important tasks and focused our attention on these areas. This section describes our experience in installing SDN equipment and the first steps to implementing protective functions.

The first priority is the ability to block traffic from an IP address. Blocking can be done by redirecting traffic to a virtual interface. In our experiments, we used two types of SDN equipment:

- A Zodiac FX board.
- A HP Aruba 2930F 4SPF Switch.

The Zodiac FX is a 4-port card that can be used as an SDN switch. The project began in 2015 through Kickstarter to provide an affordable platform for the development of software-defined networks (SDN) to developers, researchers and enthusiasts.

The Aruba 2930F switches are Layer 3 access switches. The complete set of Layer 3 functions includes support for the OSPF access layer, static and RIP routing, access control lists, sFlow and IPv6 without the need to purchase software licenses. Thus, it is potentially possible to write modules to implement all the security mechanisms described in Section II of this paper. For this experiment, a SDN segment was built. The test environment includes the Zodiac FX SDN switch and 4 hosts directly connected to it via a network interface (one per port on the board).

Each host has its own role among them, and we distinguish the following (Fig. 9):

- Web Server (192.168.0.1) — a host with the Appache and nginx web servers running on it, as well as a deployed site on the popular php Joomla framework.
- Attacker One (192.168.0.2) and Attacker Two (192.168.0.3) are malicious hosts. During the experiment, malicious DDOS traffic is sent to their Web servers from their addresses.
- Controller (10.0.1.8) — the host on which the SDN controller is deployed.

All hosts are running the Linux operating system Ubuntu.

The schematic diagram for testing the HP Aruba switch is different in that the SDN controller was OpenDayLight (ODL). At the time of this writing, we learned how to manage the network using a graphical interface and manually block traffic from IP addresses.

## VI. Conclusions and future plans

In this paper, we tried to describe the different types of network protection infrastructure. The basis for detecting intrusions is rank distributions. With their help, classification signs of network attacks were found for both the network as a whole and for IP addresses that were sources of attack.

We have described three types of network monitoring systems. The first type analyzes the traffic generated by external IP addresses. By examining the time dependence of the largest value for a network variable, we can find the threshold value for a given variable. This value will never be exceeded during normal network operation. During the attacks, the threshold value will be exceeded.

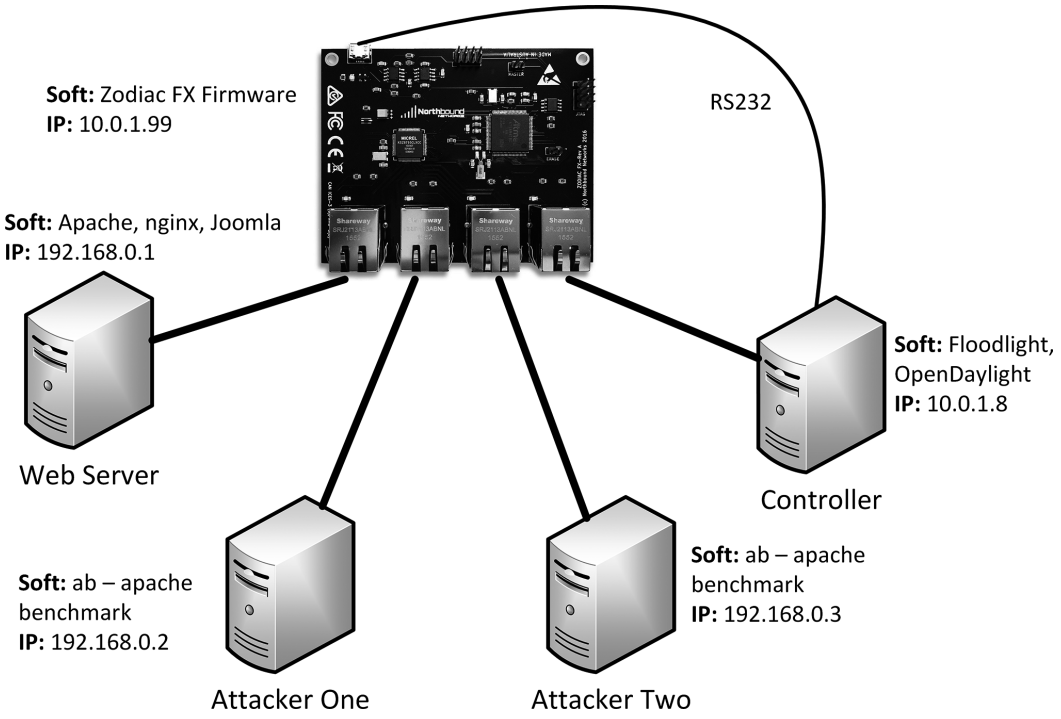


**Fig. 9.** The scheme of the experiment

The second monitoring system describes the algorithm for compiling a list of users. It consists of those addresses that form a regular service audience. After the attack begins, this list is loaded and only members of this list are allowed access to the protected resource.

Another method of preventive intrusion prevention is implemented with the help of honeypot servers. These servers are anonymously placed on the Internet and collect data on all attempts to access them. This data is then processed and classified. As a result, a list of attacking IP addresses appears in the binding to services. A portrait of the attack is also drawn up, which includes a ranked list of threats. Based on this list, it is necessary to eliminate possible vulnerabilities of information systems.

However, a full-fledged protective infrastructure is not possible without a complex coordination of efforts with providers. For such coordination, the use of SDN technologies is suggested. The article discusses the concept of a protective mechanism based on SDN and the implementation of software modules, including the interaction of controllers of different providers.

In the future, our goal is to introduce the elements of SDN technologies into the protective mechanisms we developed earlier. Attention will be paid to the model of interaction between two controllers of different providers. With this interaction, we can transfer data about attacking addresses to top-level providers so that they can redirect attacking traffic to virtual interfaces. This task must be solved to successfully counter the attack aimed at overflowing the external channels of the protected resource.

## VII. Acknowledgements

The work falls within the public tasks of the Ministry of Education and Science of the Russian Federation (2.974.2017/4.6) and was carried out with the support of grant RFBR16–07–00218a.

## References

1. M.-Y. Huang, R. J. Jasper, and T. M. Wicks, "A large scale distributed intrusion detection framework based on attack strategy analysis," Computer Networks, vol. 31, no. 23-24, pp. 2465–2475, 1999.
2. B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks," in Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks. ACM, 2010, p. 19.
3. "Dod is gearing up for the 'terabyte of death'," MeriTalk, 2018. [Online]. Available: https://www.meritalk.com/articles/dod-is-gearingup-for-the-terabyte-of-death/
4. J. Mirkovic and P. Reiher, "A taxonomy of ddos attack and ddos defense mechanisms," ACM SIGCOMM Computer Communication Review, vol. 34, no. 2, pp. 39–53, 2004.
5. C. Douligeris and A. Mitrokotsa, "Ddos attacks and defense mechanisms: classification and state-of-the-art," Computer Networks, vol. 44, no. 5, pp. 643–666, 2004.
6. J. Jiang and S. Papavassiliou, "Detecting network attacks in the internet via statistical network trac normality prediction," Journal of Network and Systems Management, vol. 12, no. 1, pp. 51–72, 2004.
7. S. Glassman, "A caching relay for the world wide web," Computer Networks and ISDN systems, vol. 27, no. 2, pp. 165–173, 1994.
8. S. A. Krashakov, A. B. Teslyuk, and L. N. Shchur, "On the universality of rank distributions of website popularity," Computer Networks, vol. 50, no. 11, pp. 1769–1780, 2006.
9. S. N. Dorogovtsev and J. F. Mendes, Evolution of networks: From biological nets to the Internet and WWW. OUP Oxford, 2013.
10. J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: learning to detect malicious web sites from suspicious urls," in Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009, pp. 1245–1254.
11. J. Markoff, "Before the gunfire, cyberattacks," New York Times, vol. 12, pp. 27–28, 2008.
12. A. Sukhov, E. Sagatov, and A. Baskakov, "Analysis of internet service user audiences for network security problems," in Telecommunication Technologies (ISTT), 2014 IEEE 2nd International Symposium on. IEEE, 2014, pp. 214–219.
13. C. Stoll, The cuckoo's egg: tracking a spy through the maze of computer espionage. Simon and Schuster, 2005.
14. L. Spitzner, "The honeynet project: Trapping the hackers," IEEE Security & Privacy, vol. 99, no. 2, pp. 15–23, 2003.
15. N. Provos, "Developments of the honeyd virtual honeypot," 2005. [Online]. Available: http://honeyd.org
16. "ab – apache http server benchmarking tool – apache http server version 2.2," The Apache Software Foundation. [Online]. Available: http://httpd.apache.org/docs/2.2/programs/ab.html
17. "Loic — free security & utilities software downloads at sourceforge.net," Dice Holdings, Inc. [Online]. Available: http://sourceforge.net/projects/loic/
18. M. Goldstein, "bonesi - bonesi - the ddos botnet simulator." [Online]. Available: https://code.google.com/p/bonesi/
19. A. Sukhov, E. Sagatov, and A. Baskakov, "Rank distribution for determining the threshold values of network variables and the analysis of ddos attacks," Procedia Engineering, vol. 201, pp. 417–427, 2017.