Intrusion detection in software defined networks with self-organized maps

Article · January 2015

CITATIONS
19

READS
1,939

2 authors, including:

Damian Jankowski
Military University of Technology
5 PUBLICATIONS **43** CITATIONS

SEE PROFILE

Paper

Intrusion Detection in Software Defined Networks with Self-organized Maps

Damian Jankowski and Marek Amanowicz

Institute of Telecommunication, Faculty of Electronics, Military University of Technology, Warsaw, Poland

Abstract—The Software Defined Network (SDN) architecture provides new opportunities to implement security mechanisms in terms of unauthorized activities detection. At the same time, there are certain risks associated with this technology. The presented approach covers a conception of the measurement method, virtual testbed and classification mechanism for SDNs. The paper presents a measurement method which allows collecting network traffic flow parameters, generated by a virtual SDN environment. The collected dataset can be used in machine learning methods to detect unauthorized activities.

Keywords—IDS dataset, machine learning, metasploit, network security, network simulation, open flow, virtualization.

1. Introduction

The Software Defined Networks (SDNs) allow to implement and control network functionality through software. Such an approach allows to deploy new services and applications in a virtual environment and to share resources with the performance and isolation from the rest of processes [1]. However, with the flexibility and scalability provided by SDNs, it is important to maintain an adequate security level. The intrusion detection technologies, integrated with the SDNs environment, can provide an additional security element, besides the classical Intrusion Detection System (IDS) and Intruder Prevention System (IPS). SDN architecture creates new opportunities to increase the security level, especially in the context of the unauthorized activities detection. Nevertheless, there are certain risks associated with this technology.

2. The Software Defined Network Technology

The basic idea of the SDNs is the separation of data plane from the control plane. In contrary to the classical network solutions, the network devices are here supervised by SDN controllers in a centralized manner. Such a solution enables configuration and programming from a host to match the service requirements in a distributed network. Moreover, the centralized logic and management allows for comprehensive monitoring network [1].

SDNs are associated with Network Function Virtualization (NFV). The current rapid development of hardware server platforms gives sufficient performance of services operating on virtual machines. Servers became more efficient and have better functionality than previously, and are suitable for use in a virtual environment. NFV is a network architecture that implements virtualization of network nodes. It enables for the functionality implementation based on the available servers, switches, storage devices, without using dedicated hardware devices. To sum up, the functionality of network hardware devices can be implemented in software technologies [2].

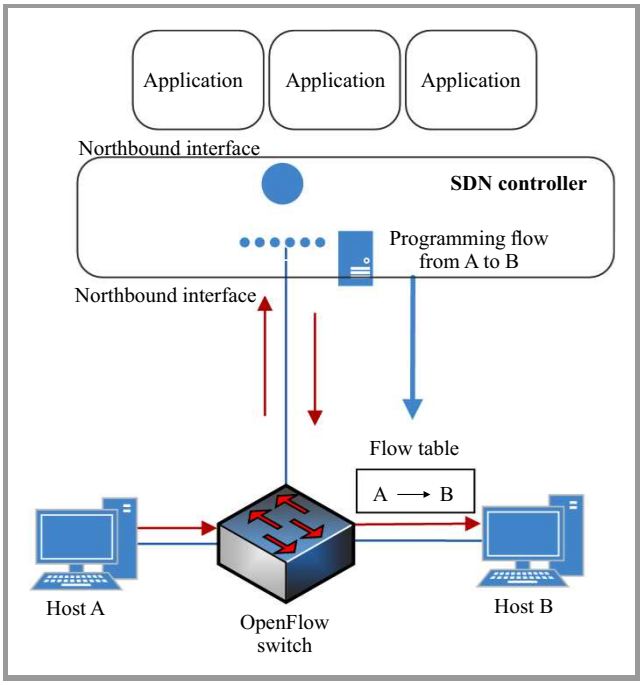

Fig. 1. Packets forwarding scheme in SDNs.

The SDN controller communicates with network devices using the OpenFlow protocol and controls network traffic according to the programmed rules. The forwarding packets methods are defined in a flow table, which is stored in SDN controllers and in switches memory supporting the OpenFlow protocol. The operation order, which describes how SDN controllers set the traffic flow, is presented in Fig. 1. In the case, that packet is forwarded to the switch

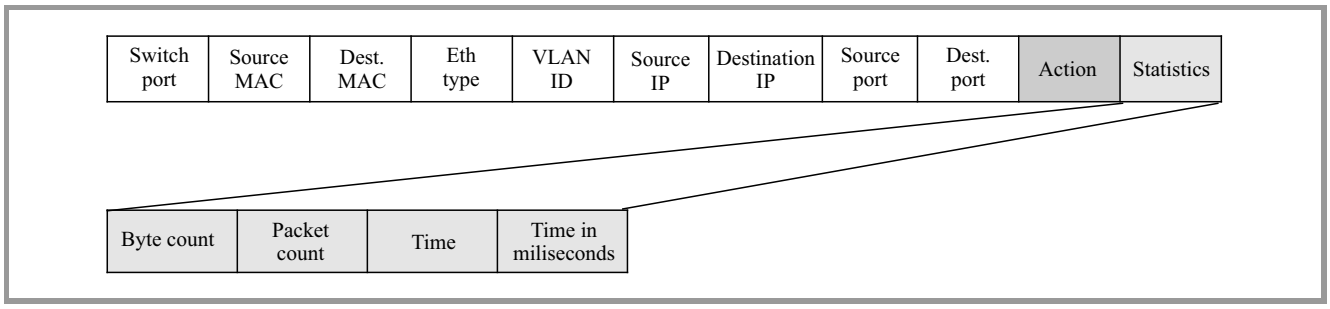

Fig. 2. Flow entry structure in SDNs.

and there is no entry in the flow table, the packet is transmitted to the SDN controller. Applications and modules, which run on controller, determine manner of packet processing. After that, the controller transmits the new entry in the flow table to the switch. That network traffic flow is then defined and established using algorithms developed on application modules in SDN controller. The structure of the traffic flow is shown in Fig. 2.

The main flow element specifies the parameters that are taken into consideration in the process of matching packets to the flow. An action field defines a way of forwarding that can be performed on packets from the particular flow. In addition, flows are linked with specific network traffic statistics, which latter can be used for traffic features extraction [2].

Thanks to the open architecture, the network logic is established with application modules running on the SDN controller. Hence, it is possible to develop algorithms fulfilling specific user functionality. Programmers can use classical programming languages, frameworks, frameworks, APIs and libraries for the process of developing application in the SDN environment.

3. Vectors of Attacks in SDNs

The SDN architecture has an important impact on the class of attack that can be performed, as presented in Fig. 3. The most dangerous situation takes place when the SDN controller is compromised. This can be done by the exploitation of vulnerabilities of processes and services running on the controller. Consequently the entire SDN domain is compromised, and the attacker has the ability to take control of all network devices. The degree of vulnerability of such attacks mainly depends on the hardware implementation of the SDN controller, programming languages and libraries used. The threats prevention can utilize IDS and IPS techniques, as well as methods of replication and recovery status of SDN servers from time before attack. Due to the nature of SDN technologies, classical IDS may be insufficient [3].

Potential security vulnerability may also exist on the administrative station, which is used to manage network operating system (SDN controller). Such terminals are used for developing applications for the control logic of network devices and ensure the monitoring of activities in the network

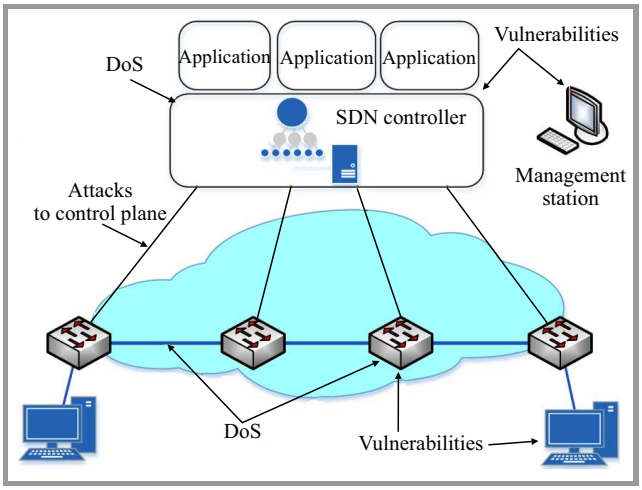

Fig. 3. Potential vector of attacks in SDNs.

environment. Attackers could potentially exploit vulnerability at the supervisor station or its connection with the controller. Reducing the risk can be achieved by making use of mutual SDN server authentication and the terminal or IDS and/or IPS techniques [4].

Other threats are attacks on the communication stream between data and control plane. For instance, the security protocol between controllers and network devices can be TLS. A potential vector of attack would exploit vulnerabilities in its implementation.

The classical network threats are still present in the SDN technologies. It is possible to generate malicious activities in IT systems, for instance deny of services or exploit vulnerabilities on servers, host or network devices [5].

4. Attacks Detection in SDNs

Despite the security vulnerabilities, SDN creates new opportunities for the implementation of more effective intrusion detection methods. Moreover, it allows for the integration of threat detection methods with the SDN environment. Due to the openness of platforms supporting SDN technologies, it is possible to use existing mechanisms and protocols. An important factor associated with intrusion detection is the possibility of aggregating statistics logs from network devices memory and forwarding them to the controller. Collected parameters can be used as source data

Table 1
Selected intrusion detection methods for SDNs

Approach	Principle of operation	Extraction of attack symptoms or features	Machine learning method or logical reasoning	Detected attacks for dataset	Network traffic	Accuracy [%]	False positive [%]
Method 1	Assessment of the first packet transmitted to the SDN controller	Maximum entropy detector, TRW-CB, Rate-limiting, NETAD	None	DoS, probe	Benign – real SDN network, attacks – artificial traffic	80–90	0–70
Method 2	Evaluation of the threats level	TRW-CB, Rate-limiting	Fuzzy logic	DoS	Artificial traffic	95	1.2
Method 3	Creating profiles using sFlow and OpenFlow	TRW-CB, Entropy level	None	DDoS, worm propagation probe	Benign – real SDN network, attack – artificial traffic	100	23–39.3
Method 4	Flow statistics collection	Flow based statistics and features	Self-organized maps of Artificial Neural Network	DDoS from botnet	Artificial traffic	98.57–99.11	0.46–0.62

for intrusion detection algorithms. In recent studies, there are a few proposals to use SDN's capabilities for intrusion detection mechanism. The four sample solutions are shown in Table 1:

- Method 1 – revisiting traffic anomaly detection using software defined networking [6];
- Method 2 – a fuzzy logic-based information security management for SDNs [7];
- Method 3 – combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments [8];
- Method 4 – lightweight DDoS flooding attack detection using NOX/OpenFlow [9].

The methods listed above detect common types of malicious activities, i.e., denial of service, port scan, and attempts to propagate malicious software. However, there is no papers describing SDN solutions, which would enable the detection of more sophisticated groups of attacks. These attacks can rely on the use of vulnerabilities in sophisticated services, and one of the phases of attack is to inject a malicious code. The presented methods have a very good detection accuracy rate, but the false positive rate is poor in approaches given by method 3. In method 1 the false positive rate may vary from 0 to 70 due to the detection technique used. Hence, it is problematic to compare the effectiveness of selected solutions because their performance tests were carried out in different environments, according to various methodologies and using different data sets.

5. The Architecture of the Proposed Approach of Intrusion Detection

The presented idea is based on the assumption that it is possible to classify whether network traffic flows represent normal operation or attack (Fig. 4). The flows classification is based on features obtained through the functionality

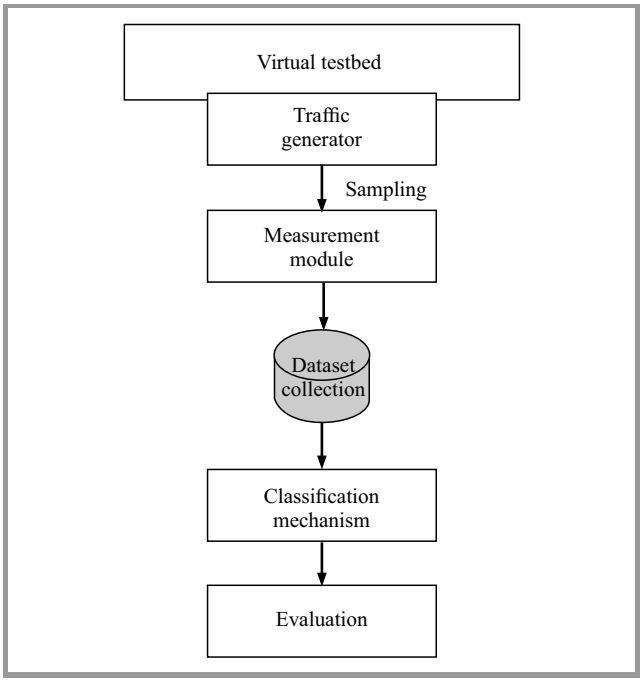

Fig. 4. Architecture of presented mechanism.

available in the SDN technology. The virtual testbed is used for generating certain classes of traffic, benign and malicious. Generated network traffic is sampled by the measurement module implemented as OSGi bundle and resides on the Opendaylight SDN controller memory. The module also programs flows. At the same time, the REST client communicates with the SDN controller and collect statistics related with flows. The present mechanism operates in the control plane. The collected data can be stored in a database or an external file. At this stage, the features for further classification can be calculated and extracted. The collected data are then converted to a dataset for testing machine-learning methods. At the current stage of research, the classification process is performed by self-organized maps of Artificial Neural Network (SOM ANN), but in future, studies would be extended to other classification methods.

6. Principle of Measurements

The measurement software module works on the SDN Opendaylight controller as an OSGi bundle [10] and implements the switch functionality (Fig. 5). The traffic flows are matched by the following criteria:

- destination IP address,
- source IP address,
- destination port of transport layer,
- source port of TCP/UDP layer,
- protocols – ARP, IP, TCP, UDP or unknown.

For each flow, an idle timeout parameter is set that defines the period after the entries are deleted from the flow table, and an identification number. Such matching network traffic distinguishes traffic in the context of different port numbers. As a result, it is possible to measure the parameters and define relationships between connections at the transport layer, which are refreshed within a specified period.
Another component used in the measurements is a REST client. This module communicates with the Opendaylight server. The following parameters permit to change the resolution of measurements:

- time between queries (in the REST client),
- time between refresh of array status and statistical parameters defined configuration flow controller SDN.

For each collected flow, a set of parameters is determined. These data values constitutes the input vector for the machine learning method:

- the measurement results contain value of n vectors features $X_i(x_1, x_2, \ldots, x_n)$ at i time of sampling. The vector features x_i are parameters and statistics retrieved from network flows;
- the input vector $X(\max\ x_1,\ \max\ x_2, \ldots, \max\ x_n)$ includes maximum values of features from all X_i vectors;
- labels defining classes are assigned to vectors X, due to IP host address.

In a review of the existing solutions in threat detection technology, SDN indicates that the described method shall detect the attack time from flow table. The presented approach is based on the flows classification by the transport layer level discrimination. It allows identifying a specific connection representing the unauthorized action. The Opendaylight software environment enables the measurement of selected parameters, which can be potentially used as features for threat detection methods.
The primary parameters obtained from the OpenFlow protocol are IP addresses, port numbers, duration, number of packets and bytes in the flow. More, the collected statistics

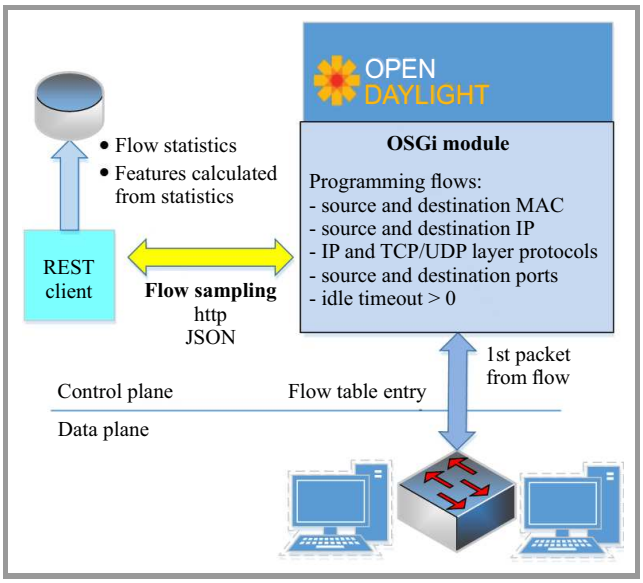

Fig. 5. Measurement method based on SDNs.

can be used to extract other information. Such values are calculated in the context of the entire array, for the sample time stamp. It reflects the dependence of connections between hosts. Example additional features are:

- single flow coefficient,
- flow rate to the host,
- multiple flows with the same host coefficient,
- flow rate of the same service to a host from multiple hosts.

The primary and additional parameters represent the x features of the input vector X. Features can be determined based on the first packet transmitted to the controller. However, at this stage of research, this mechanism is not implemented. The approach of analyzing the header and the first package contents may have a positive influence on the detection performance of attacks on specific groups. However, it is necessary to improve the performance without packets inspection, because packets can be obfuscated. Therefore, with the development of the presented method, it will be evaluated which feature is more important.

7. Virtual SDN Testbed

The most of the machine learning methods for intrusion detection is based on the KDD99Cup dataset. It is used in the process of learning and testing, allowing comparing the performance of different methods. Unfortunately, there are no such datasets that could be used to evaluate the detection methods. The proposed concept is based on the mechanism of SDN flow classification. Therefore, an important component of the presented approach is the test environment for the generation of SDN network traffic, allowing verifying the effectiveness of the presented method. In this

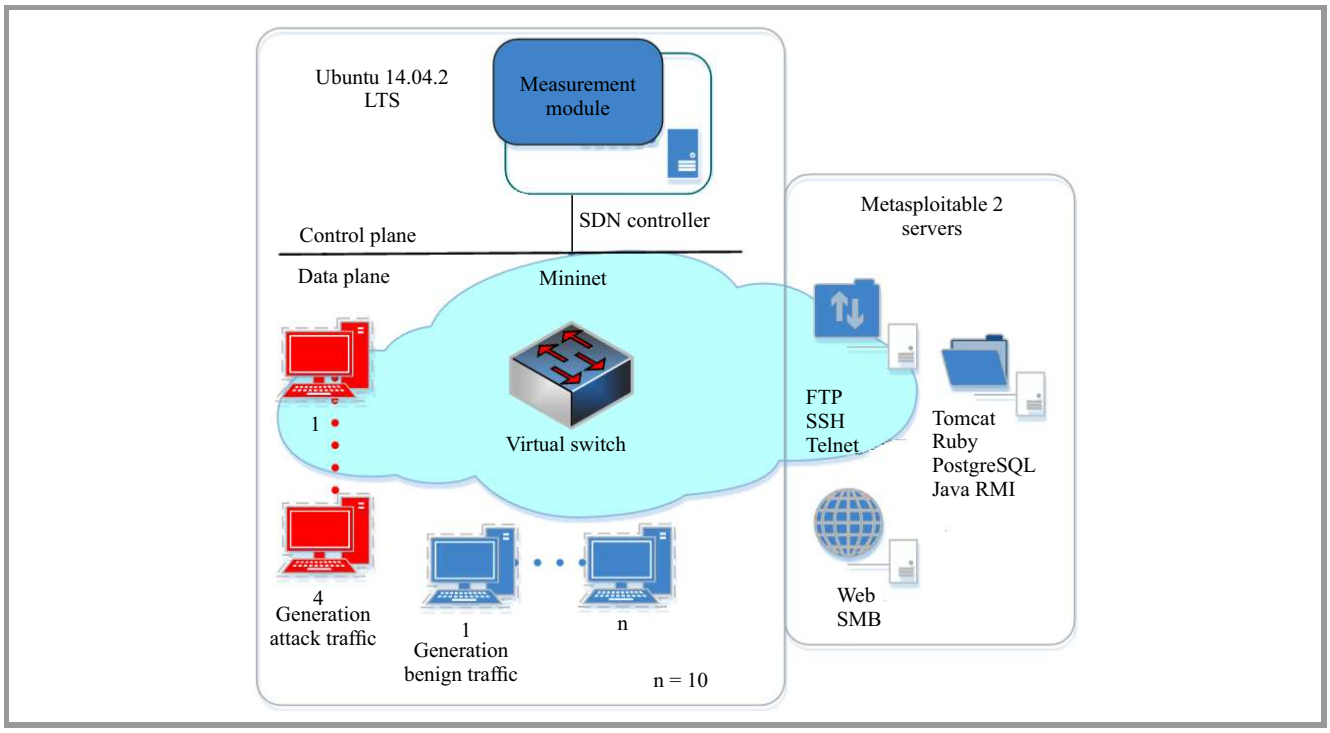

Fig. 6. Virtual SDN testbed architecture.

Table 2
Traffic classes which are performed in virtual testbed

Activities	Examples of activities	Tools for traffic generation
DoS	Denial of service	metasploit, hping3, nping
Probe	Port probe, vulnerability scan, version scan	metasploit, nmap
User2Root	Exploit vulnerabilities, shell control, backdoor	metasploit
Remote2Local	Password crack	metasploit, hydra
Normal	Communication between clients and servers	FTP, SSH, SMB, Apache, Web, Tomcat, RMI Ruby, Java RMI, Postgres, Telnet

research a SDN network emulator mininet, developed in Python and C is used. In the connection with the Opendaylight controller functionality, it is possible to model the SDN environment with different network topologies. At this stage of the implementation, the SDN has a star topology with a single switch. The architecture of the testbed is shown in the Fig. 6. The testbed covers normal traffic generation and selected groups of attacks using tools presented in Table 2.
The following classes of activities are generated in this virtual environment:

- normal – benign traffic generated between hosts and servers;
- DoS – denial of service attacks, performed against the transfer, network or computation resources of IT system;
- Probe – port, version or vulnerability scanning. Such activities give information to intruders about the potential targets of the attacks;
- U2R – attacks work by exploitation of vulnerability;
- R2L – this class covers credentials guessing and unauthorized access to IT resources.

The server side is emulated by metasploitable virtual machines with the Ubuntu Linux operating system. Vulnerabilities of services and the OS are intentionally left on the server environment. Simultaneously, the clients generate requests to the server. At the same time, the malicious host performs unauthorized activities directed to servers by using attack tools. The client activities are automated by Python scripts [11]. Generated traffic is probing by the measurement module. The servers reside on separate virtual machines and clients are virtualized on the mininet OS level.

8. Self-organizing Maps as Machine Learning Attack Detector

Machine learning methods are commonly researched concerning intrusion detection mechanisms [12]. The presented approach would use self-organized maps (SOM) to perform the unauthorized activities detection. It is a method of unsupervised machine learning, based on artificial neural

networks, useful for a graphical representation of datasets. However, when input vectors are labeled, this method can be used as a classification mechanism. The Kohonen algorithm is used as SOM learning method. As a result of the applied input signals, network indicates the activation of neurons in varying degrees, as a result of adaptation to changes in the synaptic weights, during the process of learning. Some neurons, or groups of neurons, are activated in response to stimulation, adapting to the form of specific patterns. Because of this, the test vectors activate neurons of a trained network, which are the most similar. After the initialization process, networks are reliant based on the parameters, each of the neurons is assigned to a specific position of the multidimensional space. Distribution of neurons can be created in a random way. They are associated with neighbors in a hexagonal manner. In a further stage, the network is learned by a training set. The most stimulated neuron and neighboring neurons update the weights, in response to learning vectors using the scheme (see Fig. 7):

$$W_i(k+1) = W_i(k) = \eta_i G(r)\left[X - W_i(k)\right], \qquad (1)$$

where: X – input vector of features, W_i – i weight vector of the neuron at k time, η_i – learning rate, $G(r)$ – neighborhood function given by Eqs. (3) and (4).

The distances of input vector $X(x_1, x_2, \ldots, x_j)$ to winner neurons $W(w_1, w_2, \ldots, w_j)$ are calculated on base of the Euclidean distance:

$$d(X, W_i) = \left\|X - W_i\right\| = \sqrt{\sum_{j=1}^{N}(x_j - w_{ij})^2}, \qquad (2)$$

where: X – input vector of features, x_j – j feature in X input vector, W_i – i weight vector of the neuron, w_{ij} – j value of weight in i weight vector of the neuron.

The weight adaptation degree $G(i)$ of winner and neighbors neurons is calculated by Gaussian formulas:

$$r = d(i, W) = \left\|i - W\right\| = \sqrt{\sum_{j=1}^{N}(i_j - w_j)^2}, \qquad (3)$$

$$G(r) = \mathrm{e}^{\frac{-r^2}{2\lambda^2}}, \qquad (4)$$

where: r – Euclidean distance of i neuron from winner neuron, W – winner neuron, λ – neighborhood radius.

The SOM network allows creating a type of structure, which can be represent input vectors in the best way [13], [14], [15]. It can be said, that the single neuron represents many vectors from the dataset. The class is assigned to the neuron with the consideration of which class is the most numerous, from stimulating vectors. The classification step is preformed after learning. This involves determining which labeled neuron is activated under the input vector. The dataset is normalized in the range $0 \ldots 1$. The SOM input vector records are collected data in the measuring module with assigned labels specifying the

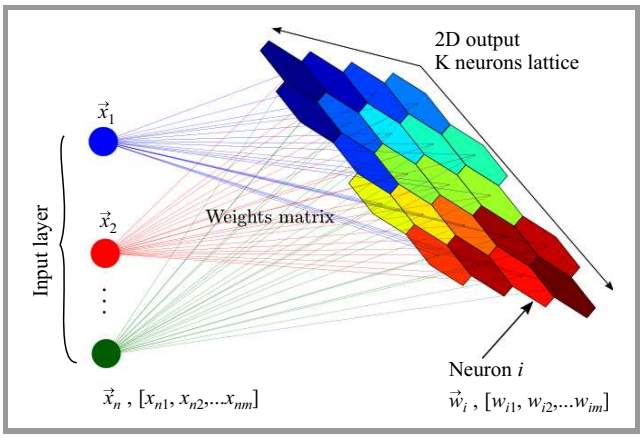

Fig. 7. Self-organizing map structure.

type of traffic. For instance, the 11 dimensional input vector $X(x_1, x_2, \ldots, x_{11})$ elements can be represented by following features:

- x_1 – destination IP address,
- x_2 – source IP address,
- x_3 – destination TCP/UDP port,
- x_4 – source TCP/UDP port,
- x_5 – duration,
- x_6 – number of packets in flow,
- x_7 – number of bytes in the flow,
- x_8 – single flow rate,
- x_9 – flow rate to the host,
- x_{10} – multiple flows rate with the same source host,
- x_{11} – rate of connections on the same port to a host from other computers.

In addition, the feature vector is linked to a label that defines the class of activities. Due to research scenario, the composition of features in input vector can vary. To evaluate performance, the cross validation method with 10 folds will be used. After the process of learning, neural network can be presented in low dimension space by using Sammon mapping. The results of the classification will be evaluated by the confusion matrix, ROC curves and typical coefficients used in machine learning methods evaluation process.

9. Summary

The presented research describes the intrusion detection method integrated with the SDN controller. This conception classifies unauthorized activities performed in SDN environment. Further studies cover the implementation of all modules and performance tests of the detection mechanism. Realized implementation and evaluation of the effectiveness will be described in further publications. In the case that the proposed mechanism will not be effective, it is necessary to research and implement additional features, especially based on the parameters and data of

the first packet in the flows. Another important stage is the research on the most significant feature selection and features extraction.

The aim of the work is also the investigation of the ability to detect a wider range of network attacks, especially those that are not identified in other technologies. It is important to study new classes of attack specified for the SDN environment. This aspect especially includes attacks on SDN controllers and the control plane. The proposed method may have potential performance disadvantages because of the high-grained traffic matching in a flow table. The use of the measuring module developed for all controllers in the network can be problematic in terms of performance. Therefore, an exemplary architecture can assume that only selected SDN controllers in the network will implement functions of intrusion detection. Another point is the comparison of other machine learning methods.

References

[1] J. Kleban and M. Puciński, "Sieci sterowane programowo SDN w centrach danych SDDC" (SDN network software controlled data centers SDDC), in *XVII Poznań Commun. Worksh. PWT 2013*, Poznań, Poland, 2013 (in Polish).

[2] D. Kreutz *et al.*, "Software Defined Networking: A Comprehensive Survey", *Proc. of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[3] S. Shin and G. Gu, "Attacking software-defined networks: A first feasibility study", in *Proc. 2nd ACM SIGCOMM Worksh. Hot Topics in Softw. Def. Netw. HotSDN'13*, Hong Kong, China, 2013, pp. 165–166.

[4] D. Kreutz, F. M. V. Ramos, and P. Verissimo, "Towards secure and dependable software-defined networks", in *Proc. 2nd ACM SIGCOMM Worksh. Hot Topics in Softw. Def. Netw. HotSDN'13*, Hong Kong, China, 2013, pp. 55–60.

[5] V. Tiwari, R. Parekh, and V. Patel, "A survey on vulnerabilities of openflow network and its impact on SDN/Openflow controller", *World Academic J. Eng. Sci.*, vol. 1, no. 01:1005, 2014.

[6] S. Akbar Mehdi, J. Khalid, S. A. Khayam, "Revisiting traffic anomaly detection using software defined networking", in *Recent Advances in Intrusion Detection*, R. Sommer, D. Balzarotti, and G. Maier, Eds. *LNCS*, vol. 6961, pp. 161–180. Berlin Heidelberg: Springer, 2011.

[7] S. Dotcenko, A. Vladyko, and I. Letenko, "A fuzzy logic-based information security management for software-defined networks", in *Proc. 16th Int. Cof. Adv. Commun. Technol. ICACT 2014*, Pyeongchang, South Korea, 2014, pp. 167–171.

[8] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, "Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments", *J. Comp. Netw.*, vol. 62, pp. 122–136, 2014.

[9] R. Braga, E. Mota, and A. Passito, "Lightweight DDoS flooding attack detection using NOX/OpenFlow", in *Proc. 35th Ann. IEEE Conf. Local Comp. Netw. LCN 2010*, Denver, Colorado, USA, 2010, pp. 408–415.

[10] OpenDaylight Platform [Online]. Available: https://www.opendaylight.org/

[11] Mininet – An Instant Virtual Network on your Laptop (or other PC) [Online]. Available: http://minimet.org

[12] A. S. Subaira and P. Anitha, "Efficient classification mechanism for network intrusion detection system based on data mining techniques: a survey", in *8th IEEE Int. Conf. Intell. Syhst. & Control ISCO 2014*, Coimbatore, India, 2014, pp. 274–280.

[13] K. Choksi, B. Shah, and O. Kale, "Intrusion detection system using self organizing map: a surevey", *Int. J. Engin. Res. Appl.*, vol. 4, no. 12, pp. 11–16, 2014.

[14] S. Osowski, *Sieci neuronowe do przetwarzania informacji* (*Neural Networks for Information Processing*). Warsaw: Publishing House of Warsaw University of Technology, 2013 (in Polish).

[15] "SOMz: Self Organizing Maps and random atlas" [Online]. Available: http://lcdm.astro.illinois.edu/static/code/mlz/MLZ-1.2/doc/html/somz.html

Damian Jankowski received B.Sc. and M.Sc. degrees from the Military University of Technology, Warsaw, Poland in 2010 and 2011, in Telecommunication Engineering. His research interests include programming, system virtualization, system administration, IT security, machine learning, and data mining.

E-mail: damian.jankowski@wat.edu.pl
Military University of Technology
S. Kaliskiego st 2
00-908 Warsaw, Poland

Marek Amanowicz received M.Sc., Ph.D. and D.Sc. degrees from the Military University of Technology, Warsaw, Poland in 1970, 1978 and 1990, respectively, all in Telecommunication Engineering. In 2001, he was promoted to the professor's title. He was engaged in many research projects, especially in the fields of communications and information systems engineering, mobile communications, satellite communications, antennas & propagation, communications & information systems modeling and simulation, communications and information systems interoperability, network management and electronics warfare.

E-mail: marek.amanowicz@wat.edu.pl
Military University of Technology
S. Kaliskiego st 2
00-908 Warsaw, Poland