Position papers of the 2014 Federated Conference on Computer Science and Information Systems pp. 143–148
DOI: 10.15439/2014F473
ACSIS, Vol. 3

SDN Architecture Impact on Network Security

K. Cabaj
Warsaw University of Technology
Nowowiejska 15/19
00-665 Warsaw,
Poland, Email:
kcabaj@ii.pw.edu.pl

J. Wytrębowicz
Warsaw University of Technology
Nowowiejska 15/19
00-665 Warsaw,
Poland, Email:
j.wytrebowicz@ii.pw.edu.pl

S. Kukliński
Warsaw University of Technology
Nowowiejska 15/19 00-665 Warsaw, Poland,
Email:
kuklinski@tele.pw.edu.pl

P. Radziszewski
Warsaw University of Technology
Nowowiejska 15/19
00-665 Warsaw,
Poland, Email:
pmr@ii.pw.edu.pl

K. Truong Dinh
Warsaw University of Technology
Nowowiejska 15/19 00-665 Warsaw, Poland,
Email:
k.truongdinh@stud.elka.pw.edu.pl

Abstract—The Software Defined Networking (SDN) paradigm introduces separation of data and control planes for flow-switched networks and enables different approaches to network security than those existing in present IP networks. The centralized control plane, i.e. the SDN controller, can host new security services that profit from the global view of the network and from direct control of switches. Some security services can be deployed as external applications that communicate with the controller. Due to the fact that all unknown traffic must be transmitted for investigation to the controller, maliciously crafted traffic can lead to Denial Of Service (DoS) attack on it. In this paper we analyse features of SDN in the context of security application. Additionally we point out some aspects of SDN networks that, if changed, could improve SDN network security capabilities. Moreover, the last section of the paper presents a detailed description of security application that detects a broad kind of malicious activity using key features of SDN architecture.

I. Introduction

In this paper we analyse the features of SDN that can be used for improving network security. We do not analyse security of SDN per se, however some mechanisms, that directly protect users, improve the security of the SDN network too. Additional information concerning threats, and ideas how SDN network should be secured, can be found in the Kreutz et al paper [1]. Even though the SDN concept is novel, some articles concerning detection of various kinds of known attacks are already published. Data from an SDN controller allow detection of network scans [2], [3], DoS and DDoS attacks [2], [4], and detection of infected Zombie machines that are part of a botnet [3]. Additionally, SDN networks can be easily reconfigured to pass traffic for inspection by various legacy (not SDN capable) security devices, and next automatically react on an attack detected by one of those devices. An SDN network has the ability to easily add new network functionalities. The functionalities added as specialized applications (atop or inside the SDN controller) have access to each flow forwarded by the network. Moreover, an application co-working with the SDN controller can easily add rules to SDN switches, completely changing flow switching or even changing the content of forwarded packets. The security functions are not packet-based but flow-based, what makes protection more efficient. The centralization of control plane operations gives the ability to correlate events from different network nodes, what enables a new approach to network security. In more detail we will discuss all these issues in the third chapter.

The paper is organized as follows. In the next section key features of SDN networks are presented. The third section presents the impacts of the SDN paradigm on security mechanisms implementation. Section IV describes the concept of an application that utilizes evolved SDN networks in order to achieve more efficient attack detection.

II. KEY FEATURES OF SDN

The most expected and promising aspects of SDN networks are associated with:

- centralization of some network operations that enables to base control mechanisms on global network view e.g. traffic engineering,
- easy and standard way in which applications may interact with the network via so called “North-Bound API”,
- easy customization of networks.

Open Networking Foundation (ONF) and International Telecommunication Union (ITU) have been recently working on the standardisation of SDN networks. A high level view of the SDN architecture together with the key principles of SDN networks have been presented by ONF [5]. The SDN controller acts as a network “brain” (see Fig. 1), directly communicates with network applications via North-Bound Interface (Control – Application Plane Interface) to provide network state information from data plane, and to translate requirements and high-level policies from applications to low-level commands via South-Bound Interface (Control-Data Plane Interface). The most popular protocol used today for communication between the SDN controller and network data plane is OpenFlow [6].

II. This work has been partially conducted as part of the CoSDN (Cognitive Software Defined Networks) project, which is funded by FNR Luxembourg and NCBiR Poland.

© 2014, PTI

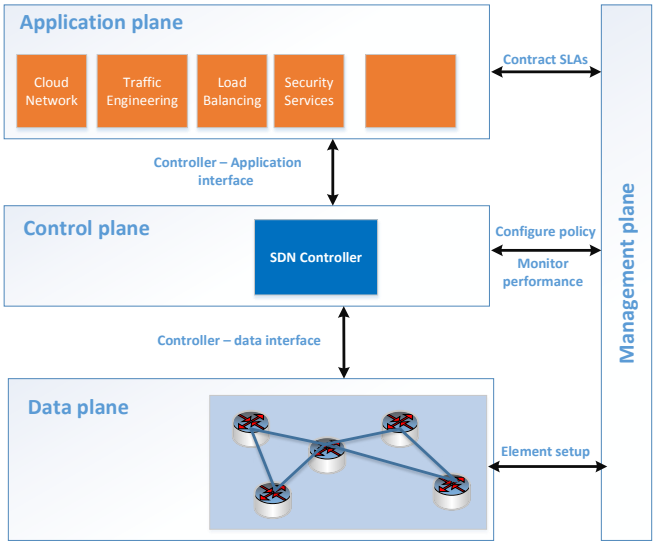

Fig. 1 SDN architecture overview

Fig. 1 describes the general SDN architecture according to network planes and interactions between them. The SDN networks are divided into Data, Control, Application and Management planes.

- Data plane consists of network forwarding elements i.e. switches, which main task is to forward incoming flows to their destinations, making use of routes defined in flow tables.
- Application plane is composed of network service applications, business services, security services, and others, which communicate with the network infrastructure through the SDN controller. They can benefit from abstracted global view of the network according to their own purposes.
- SDN controller is the central point of the network; it gives decisions to the data plane how to forward or modify flows. The controller is also responsible for the transformation of applications' commands to the lower-level communication protocol used by the data plane devices.
- Management plane (according to ONF) is responsible for tasks that are better handled outside the control, application and data planes. It should be isolated and hidden from users. Management entity handles tasks such as setting up the network or configuration of network parameters. It should not be programmable from outside, in order to prevent any kinds of network attacks and to protect the entire network.

North-Bound Interface (NBI) is the interface between applications and the controller. It provides access to network resources from the application level. Although NBI is still not defined, it may provide authorisation and authentication for applications. A role-based authorisation approach has been proposed by Porras et al. [7].

Conflicting rules from applications appear in the controller when some applications require different network behaviour. The conflicts are hard to handle due to the complexity of network control tasks, and an orchestrator is needed. The management in SDN can be implemented in the controller, in the management plane or as a separate application.

The controller is directly connected to network forwarding elements via South-Bound Interface (SBI). OF technology seems to be dominant today, it has been already deployed in many SDN networks [8].

III. IMPACT OF SDN ON NETWORK SECURITY

The SDN concept moves traditional networking from hardware to software with the benefit of automating and simplifying network operations and administration and improving the network performance. As a new technology, SDN is subject to vulnerabilities. In-line with powerful capabilities, which are introduced by SDN networks, various drawbacks of the approach also exist.

Taking into account implementation of security in SDN networks, the following things have to be considered:

- Interactions with network nodes (switches) are performed via the OpenFlow protocol, which has some limitations.
- Global network view: monitoring information is available at the controller; it has the ability to directly manipulate each flow, including possibility to kill it at the source.
- No middleboxes, including NAT (Network Address Translation) or firewalls, are defined in the architecture.

A. OpenFlow limitations in the context of security

Currently, OpenFlow is the most popular protocol used between the SDN controller and network forwarding elements. Although other SDN control interfaces protocol and languages (like FML, Procera, Frenetic [11]) had been designed, the OpenFlow protocol gained the dominant position, and it evolves enabling processing of more and more protocol headers. Despite its popularity, the protocol has some drawbacks. The first big limitation is associated with strict definition of fields that are used by forwarding rules and can be altered by the rules. For example, fields used by IPv6 protocol were not introduced until OpenFlow version 1.3. If hardware or software use older OpenFlow version, the IPv6 traffic cannot be forwarded. The possibility of altering different protocol fields, in an OpenFlow capable switch, enables more complicated functions than forwarding, e.g. NAT or firewall.

It is impossible to implement actions that use other than defined by OpenFlow packet's fields. A solution that resolves this OpenFlow limitation has been recently proposed, it is Protocol Oblivious Forwarding (POF) [12]. A field in POF is defined as sequence of bits starting from a

given offset and having a certain length. This provides a flexibility that is the main advantage of the POF paradigm. In contrast, the OpenFlow approach defines a limited number of fields that can be used during matching and altering phase. The POF flexibility allows for rapid development and implementation of new, specialized network protocols, without changes in the switch hardware and without changes in communication with the controller. Moreover, applications that implement new SDN services using POF can make decisions using any part of the ingress packet. This gives a huge flexibility and unimaginable nowadays functionality of future SDN enabled networks. Especially for security related applications, this flexibility can be beneficial, allowing implementation of a "Deep packet inspection" [12], for example finding of known exploits in analysed traffic. Further adoption of POF in network devices hardware structures can lead to performance boost, not achieved in software solutions.

B. Centralized network operations and security

From many years IP network monitoring systems were developed to gather and aggregate data from all network nodes by the use of various protocols, e.g. SNMP [9] or NetFlow [6]. However, access to data related to flow traffic, forwarded in the network, is not easy and not efficient. For example, multiple queries have to be used to gather current state of the network, which could utilize large percentage of management bandwidth and monitored device CPU. In SDN such information is easily accessible at the SDN controller, while in traditional IP networks it has to be sampled on packet basis. From the security point of view, the global awareness concerning all devices in the network is beneficial. An analysis, concerning the whole traffic observed in the network, could lead to detection of distributed attacks, what is impossible on a single network device. Examples of such threats are: a stealth scanning concerning whole network, a set of infected machines, and the advanced persistent threats (APT) [10]. The second big advantage of the global view is associated with the SDN controller's ability to manipulate each flow forwarded through the network. Any reaction to a detected threat can be immediate. From years such reactions were implemented by sending specially crafted reset packets, real-time firewall rules, or by placing security devices in the inline mode, where whole traffic is passed through it. All of those solutions are inefficient and even could lead to degradation of network performance.

The centralized architecture has also some drawbacks. The most evident is associated with performance, when vast amounts of network flows must be analysed in one place. Additionally such architecture introduces a single point of failure. This can lead to congestion of the SDN controller, when many flows are generated in short time, for example as an effect of infection or aggressive scanning attempt. Additionally, as it was described in [1], malicious users can deliberately generate fake traffic to disturb an SDN network. These observations reveal question if all security decisions should be performed directly on the SDN controller. In fact, any successful attack on a centralized controller (may it be a DoS or it's compromise) can result in severe network degradation. Logical distribution of physical controllers might alleviate this danger to some extent, but a meticulous protection of control resources is critical. The protection should cover all aspects – not only technical, but also "social". In legacy networks this kind of danger is not always critical, and impact of a single security breach can be contained. A carefully thought out network design (e.g. routing policies, OSPF areas, individual link protection) is the solution for security enforcement of today IP networks.

C. Lack of middleboxes in SDN

In currently operating networks many functionalities are implemented in the form of additional devices, so called middle boxes, e.g. NAT devices or firewalls. As was presented in the previous section, OpenFlow limitations prevent implementation of some functions, for example deep packet inspection. Moreover, it is not optimal, from performance point of view, to process all decisions concerning every flow by the SDN controller. A decentralization of some SDN functions, even though it breaks the SDN paradigm, can lead to more efficient and scalable networks. The decentralized functions can be performed locally on SDN switches. This solution demands supplying the SDN switch with an execution environment on which local applications can run. Implementation of security functions in this place has many advantages. As was proven in [14], this can improve detection rate in comparison to traffic observed in aggregated links owned by ISP. Sample description of such solution (PDEE – Programmable Distributed Execution Environment) can be found at [15].

On the other hand, lack of middleboxes in the architecture definition can imply deficiencies in security, however it facilitates end-to-end connectivity, which is needed for some network applications.

A big problem in legacy networks is how to apply and tune traffic engineering rules for tunnelled or encrypted data streams. Without auxiliary mechanisms different tunnelled flows are processed in a unified manner. Of course end devices can alter traffic policies or divide data stream into multiple tunnels in spite of changes in the network core, but only if they are aware of that fact. The SDN approach makes such operations more natural.

A specific kind of middlebox is Intrusion Detection Systems (IDS). Simple IDSs analyze signatures or anomalies, more advanced ones utilize data exploration algorithms. In the next chapter we describe an example of the IDS application, which is suitable for the specifics of SDNs.

IV. SDN SECURITY APPLICATION EXAMPLE: DISTRIBUTED FREQUENT SETS ANALYSER

As was presented in the previous sections, SDN networks enable execution of specialized software that could add new services to the existing networks. Such software could alter simple switches into powerful middle boxes or specialized security devices. What should be emphasized, such change does not need any exchange of deployed hardware and is associated only with addition of new software components. Moreover, the whole network view possessed by the controller allows implementation of advanced methods that could utilize such knowledge. In this section an idea of Distributed Frequent Sets Analyzer (DFSA) systems is presented. The DFSA system takes advantage from experiments with anomaly detection using data mining, and from the features of SDN network. In effect DFSA system could effectively detect broad range of modern network threats.

The idea of data mining algorithms usage for security is based on works [16] [17]. Unfortunately, the integration of three most important elements of such system, i.e. data gathering, data analysis and implementation of actions, is not a seamless process in existing IP networks. For this purpose various mechanisms, techniques and software modules have to be used. Transfers of information between the mentioned elements have impact on the overall security system performance. In contrast, the implementation of such functionality as a security application for SDN network should seamlessly integrate all processes needed for data acquisition, data analysis and reaction to a detected threat.

This section describes a sample security application, which utilizes the well-known features of SDN network and some emerging enhancements that in our opinion can improve the overall network security. Presented concept can be used for detection of various evil or at least anomalous activities performed by network terminals

A. The concept of frequent set analysis

It has been proven that many modern threats, when activated, produce similar patterns in observed traffic. For example network scanning, denial of service attacks (both using one machine or distributed system), botnet activity, sending spam and many more [16]. Discovery of such repeated activity can be a sign of an attack. The data mining techniques could be successfully applied to discover such patterns. Their most important advantage is associated with fact that discovered results are understandable by humans, and can be easily and automatically converted into a response to the detected threat. One of such methods that can detect so called frequent sets is described in [18]. In this method the analysed data is treated as a collection of sets, where each set represents one flow. Each set consists of individual items, which are associated with used protocol, addresses, ports, number of transmitted packets, and overall data size. The number of all sets in the analysed collection that contains this given subset is called support. A frequent set is a subset, whose support is equal or greater than minimalSupport – a parameter defined by the user. Table 1 presents a sample data set with the flows observed by an SDN controller.

TABLE I.
SAMPLE DATA SET USED IN THE EXAMPLE

	Prot.	Src IP	Src Port	Dst IP	Dst Port
1	TCP	10.1.X.X	54333	192.168.Y.Y	80
2	TCP	10.1.X.X	54333	192.168.Y.Y	80
3	TCP	10.1.X.X	54333	192.168.Y.Y	80
4	TCP	172.16.Z.Z	42356	192.168.Y.Y	80
5	TCP	172.16.Z.Z	42456	192.168.Y.Y	8080
6	TCP	172.16.Z.Z	44895	192.168.Y.Y	1080

An operator, who observes the traffic in the network, using his knowledge and experience, sets the minimalSupport parameter. Assumed that in this example we set minimalSuport to 3, various frequent sets can be detected, for example <tcp, *, *, *, * >, <tcp, *, *, *, 80>, <tcp, *, *, 192.168.Y.Y, 80>, <tcp, 10.1.X.X, 54333, 192.168.Y.Y, 80> or <tcp, 172.16.Z.Z, *, 192.168.Y.Y, *>. Asterisks represent items that do not appear in the detected frequent sets. Above frequent sets support initial item sets from table 1 in ranges 1-6, 1-4, 1-4, 1-3 and 4-6 respectively. The most comprehensible patterns are the last two, which are called maximal, due to the fact, that there are no other detected frequent sets in this data set that are supersets of them. Both of these two maximal frequent sets have the support value equal to 3. For further analysis only maximal frequent sets are considered.

Depending on items contained in a frequent set and its support a decision about corresponding flows character can be taken. For example, when detected frequent set has very high support, items are: TCP protocol, TCP port number 25 and only single source address of a desktop machine, it can be assumed with high probability that this machine is infected and actively sends spam. In the other case, when in a long period of observation, frequent set with moderate support value and items associated with TCP protocol, source port and source IP address are detected, presumption that someone used nmap, one of the most known network scanners, can be taken. Detection of such frequent set is caused by this particular scanner implementation, which during scanning uses only one source port. Such analysis can be performed on each network access node and can detect attacker or victim machine directly connected to this particular node. Moreover, the traffic patterns can be observed by a special node that has global view of the network (NetFlow collector, etc.). Such analysis can be performed in a network, which does not support any local detection (the case of existing IP routers). This approach detects scanning activity or other activities that do not appear

frequently at a single device, but appear in the whole network.

B. Implementation of DFSA in SDN

The above described method can be efficiently and easily implemented in SDN. Additionally, the ability to reconfigure flow tables in SDN switches can be used for implementation of automatic reaction to detected threats, for example dropping of offending traffic or its degradation. We proposed a hybrid approach, which is based on both local and global analysis of traffic patterns. The Local Frequent Sets Analyser (LFSA) is placed in each SDN switch, and it detects threats that generate vast amounts of traffic; for example infected machines, which send spam or perform DoS attacks. These malicious activities can be efficiently, easily, and without delay detected locally. In effect, the time from detection of an attack to the reaction to it is as short as possible. Moreover, as described in [14], the detection performed at the access node has better accuracy than this performed at aggregation links, for example at ISP management data centre. Additionally, performing all actions locally on one switch reduces the traffic, which is exchanged with the centralized traffic collector and analyser and scales better. However, not all types of attacks can be detected locally. For example stealth scanning of the whole network can be undetected on a single switch, but it can be observed having the global network view. Due to this fact the Global Frequent Sets Analyser (GFSA) is used in our approach.

GFSA is a single module that is placed in the SDN controller or in an SDN application connected to the controller via the NBI interface. LFSA should be implemented at every SDN switch. According to the SDN architecture it is now impossible. However, there exist some enhancements to the concept of SDN, which allow hybrid implementations, for example PDEE [15]. In Fig. 2 the architecture of proposed Distributed Frequent Set Analyser (DFSA) system is presented.

According to the PDEE concept the modules of the DFSA system can be implemented together with the Management applications (M), running at the PDEE execution environments. The LFSA module is placed on each SDN switch and GFSA on the SDN controller. Additionally, DFSA adds to the SDN manager a specialized console module that can be used for the modules configuration (e.g. setting of minimalSupport value used by data mining algorithms). Moreover, the console can be used for reviewing the DFSA logs, which contain information concerning all detected anomalies and performed actions.

For each flow forwarded by the SDN switch, the related set is created and passed to the LFSA. What should be emphasized, cost of data pre-processing is negligible, due to the fact that all needed information is prior collected by the SDN switch. In contrast to mentioned earlier solutions, this computation part is very CPU intensive, as all packets must be directly examined using promiscuous mode or be examined in system firewall and later the firewall processing results are logged and parsed. The process of frequent set discovery is executed at preprogrammed intervals.

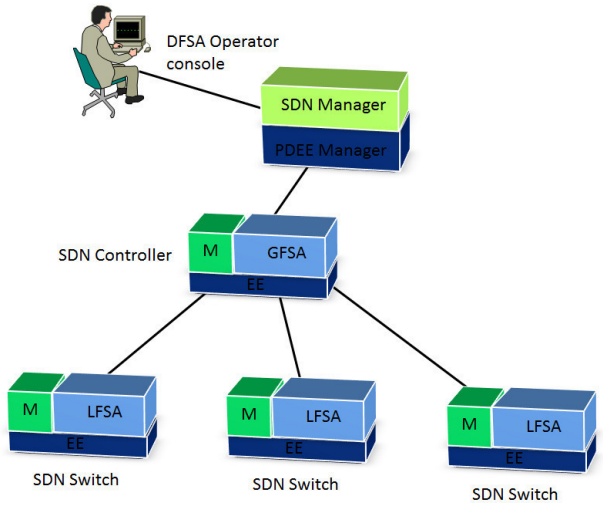

Fig. 2 Architecture of the DFSA system, using the PDEE environment.

Discovered patterns are analysed for symptoms of probable anomalous activity. As it was described in the previous paragraph, the support value of a discovered frequent set and items that the set contains can be used for this purpose. When probability of the malicious activity is very high, the corresponding flows can be stopped by using elements of the discovered frequent set (protocol, source IP, destination port, etc.). For this purpose LFSA inserts additional rules into flow tables of the SDN switch, to drop packets related to malicious flows. In case when malicious activity is not evident, the flow can be degraded and thus slow down attack, but not completely remove it – to preserve the communication under the question. This kind of functionality is nowadays used for flow control (e.g. Random Early Detection), but can be used as a security function as well. When the malicious activity is detected locally, an appropriate action is locally executed. This action not only stops the attack but also protects the SDN controller from DoS attacks directed to it. Additionally, as it has been proven in [13], these attacks can be more accurately detected in the access switch, where the offending machine is connected, rather than in aggregation switch.

At the global level (i.e. at the SDN controller) only the analysis of the aggregated data is performed. The LFSA module, executed in the SDN switch, sends to GFSA implemented in the SDN controller each set that does not appear in discovered frequent sets. This kind of filtering is based on the assumption, that all the activities that generate high volume data traffic are detected at the switch level. There is no need to detect them once again at the central level in the GFSA module. In effect, the GFSA module performs frequent set discovery using aggregated data from all switches. Moreover, data associated with high volume

attacks are filtered, and even there is no need to transmit it to the SDN controller. This approach can minimize traffic overhead and CPU cycles at the SDN controller.

Global analysis performed in the GFSA module can be beneficial for detecting massive scanning activity and some stealth scanning techniques. Due to low volume of data associated with those kinds of attacks, observed in a single switch, they cannot be locally detected. However, when data sent from all LFSA modules are received by GFSA and aggregated, detected frequent sets lead to discovery of these threats. Additionally, due to initial filtering that leads to smaller volume of data, aggregation of sets before detection of frequent sets can be performed over a longer time range. After aggregation of frequent sets a discovery is performed. Analysis of discovered patterns is similar to that at the local level. The only change is associated with the manner in which reaction is performed. In this case the SDN controller contacts with all involved SDN switches and installs appropriate rules in their forwarding tables.

V. Conclusion

The SDN paradigm on one hand simplifies the implementation of some security mechanisms, mostly due to centralization of control operations, on the other hand limits distributed approaches. The proposed DFSA system, which uses features of SDN network, can be used for efficient and reliable detection of various network attacks that are observed nowadays in IP networks. The hybrid architecture, which extends the SDN paradigm, allows fast detection of attacks that generate huge amount of traffic, directly at the SDN switch using LFSA modules. Moreover, the GFSA module executed at the SDN controller can be used for detection of attacks that concerns the whole network. Additionally, using SDN network ability to change flow tables, automatic reaction can be implemented as soon as a threat is detected. Implementation of the concept requires a modified SDN, as defined in [15].

Acknowledgment

References

[1] D. Kreutz, F. Ramos, and P. Verissimo, “Towards secure and dependable software-defined networks,” Proceedings of the second ACM SIGCOMM workshop on “Hot topics in software defined networking,” pp. 55-60, 2013, http://dx.doi.org/10.1145/2491185.2491199.

[2] S. A. Mehdi, J. Khalid, and S. A. Khayam, "Revisiting traffic anomaly detection using software defined networking," Recent Advances in Intrusion Detection. Springer Berlin Heidelberg, 2011, http://dx.doi.org/10.1007/978-3-642-23644-0_9.

[3] S. Shin, et al. "Fresco: Modular composable security services for software-defined networks," Internet Society NDSS, 2013.

[4] R. Braga, M. Edjard, and P. Alexandre, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," Local Computer Networks (LCN), 2010 IEEE 35th Conference on. IEEE, 2010, http://dx.doi.org/10.1109/LCN.2010.5735752.

[5] Open Netwok Foundation, “SDN Architecture Overview,” version 1.0, 2013.

[6] N. McKeown, T. Anderson, H. Balakrshman, G. Parulkar, L. Peterson, J. Rexford, S. Shenker and J. Tuner, “OpenFlow: enabling innovation in campus networks,” Sigcomm Comput. Commun., vol. 38, no. 2, pp. 69-74, 2008, http://dx.doi.org/10.1145/1355734.1355746.

[7] Porras, Philip, et al. "A security enforcement kernel for OpenFlow networks," Proceedings of the first workshop on “Hot topics in software defined networks,” ACM, 2012, http://dx.doi.org/10.1145/2342441.2342466.

[8] S. Jain, et al. "B4: Experience with a globally-deployed software defined WAN," Proceedings of the ACM SIGCOMM 2013 conference, http://dx.doi.org/10.1145/2486001.2486019.

[9] J. Case, "A Simple Network Management Protocol (SNMP)," IETF RFC1157, 1990.

[10] C. Tankard, “Advanced Persistent Threats and how to monitor and deter them,” Network security, 2011, http://dx.doi.org/10.1016/s1353-4858(11)70086-1.

[11] A. Doria, J. Salim, R. Haas, H. Khosravi, W. Wang, L. Dong, R. Gopal and J. Halpern, “Forwarding and Control Element Separation (ForCES) Forwarding Element Model,” IETF, 2010.

[12] H. Song, “Protocol oblivious forwarding: Unleash the power of SDN through a future proof forwarding plan,” Sigcomm HotSDN workshop, 2013, http://dx.doi.org/10.1145/2491185.2491190.

[13] A. Nakao, “Deeply programmable network: Emerging technologies for network virtualization and Software Defined Networks,” ITU-T Kaleidoscope, Kyoto, 2013.

[14] S. A. Mehdi, J. Khalid, S. A. Khayam, “Revisiting Traffic Anomaly Detection using Software Defined Networking,” Recent Advances in Intrusion Detection Lecture Notes in Computer Science Volume 6961, 2011, pp. 161-180, http://dx.doi.org/10.1007/978-3-642-23644-0_9.

[15] S. Kukliński, “Programmable Management Framework for Evolved SDN,” IEEE/IFIP Network Operations and Management Symposium, Poland, 2014.

[16] K. Cabaj, K. Szczypiorski, S. Becker, “Towards Self-defending Mechanisms Using Data Mining in the EFIPSANS Framework,” Advances in Multimedia and Network Information System Technologies, Advances in Intelligent and Soft Computing, nr 80, 2010, Springer, pp. 143-151, http://dx.doi.org/10.1007/978-3-642-14989-4_14.

[17] K. Cabaj, Z. Kotulski, P. Szałachowski, et al., “Implementation and testing of Level 2 security architecture for the IIP System,” Przegląd Telekomunikacyjny - Wiadomości Telekomunikacyjne, SIGMA NOT, vol. LXXXV, nr 8-9/2012, 2012, pp. 1426-1435.

[18] R. Agrawal, T. Imielinski, A Swami, “Mining Association Rules Between Sets of Items in Large Databases,” Proceedings of ACM SIGMOD Int. Conf. Management of Data, 1993, http://dx.doi.org/10.1145/170036.170072.