Covenant Journal of Informatics & Communication Technology. Vol. 6 No. 2, Dec., 2018

An Open Access Journal Available Online

Security Algorithm for Preventing Malicious Attacks in Software Defined Network (SDN)

Oluwasogo Adekunle Okunade [1], OluwaseyiOsunade [2] & Emmanuel GbengaDada [3]

[1] Department of Computer Science, Faculty of Sciences,
National Open University of Nigeria, Abuja, Nigeria.
[2]Department of Computer Science, University of Ibadan,
Ibadan, Nigeria.
[3]Department of Computer Engineering, Faculty of Engineering,
University of Maiduguri, Nigeria.

aokunade@noun.edu.ng1, seyiosunade@gmail.com[2],
gbengadada@unimaid.edu.ng[3]

Abstract—This paper explores the success record of the Internet as well as its shortcoming in the area of network configuration, response to fault(s), load and change(s) that led to the concept of Software Defined Network (SDN).These are the factors that separated combined network's control from forwarding planes for easier optimization, programming of network and centralization of control logic capabilities. These had also led to new different challenges, that open doors for new threats that were not existing or harder to exploit. SDN prototype embraces third-party improvementas a result of hard work, that later makes the SDN vulnerable to potential trust issue on its applications (apps).This makes it possible for an intruder toinsert malicious content/programs into the network packets and then forward into the network.Codes were written to implement the designed algorithm using white/blacklist source identification combined with Hash Bayes' Theorem (W/B+HBT) content filter as a security measure to prevent the malicious attack(s). It was shown that new transaction(s) from known attack source(s) are classified as Blacklist and dropped, while those known as whitelist are forwarded to their respective destination as a legitimate packet(s) (W/B). Those from unknown sources were treated using Hash Bayes' Theorem (HBT) content filter. The result of the implementation is able to record 10% false

positive (FP) and false negative (FN) and 90% true positive (TP) and true negative (TN) (accurate classification of packets) for the presented algorithm.

Keywords/Index Terms— OpenFlow, Flow table, Control plane, Hash Bayes' Theorem, Security Algorithm

1. Introduction

Software Defined Network (SDN)is anemerging innovative technology for enabling open programmable network environment to realize network with efficient and dynamic nature. it isdynamic, manageable, inexpensive network components and high-speed network emerging services according (Yutaka, Hung-Hsuan&Kyoji, 2013 and Raphael, Dietmar&Mark, 2015).Before the advent of dynamic nature SDN, the complexities of traditional computer networks were being managed with theadding of more protocols suites to meet up with the required expectation despite its complexity according to (Muhammad *et al.*,2014).Open Networking Foundation (ONF) is a profitless organization dedicated to the development,standardization, and commercialization of SDN according to (Wenfeng*et al.*, 2015). However, the openness of the SDN has resulted in security challenges that could jeopardize its purpose of existence if left unaddressed. This had made security a major concerned for SDN, as a result of its distinguishing features, conventional network security approaches cannot be directly applied. For the fact thatSDN improves network performance, yet it creates some peculiar challenges due to its centralized control and programmability features. It introduces security control challenges(Diego *et al.*,2013; Phillip*et al.*, 2012; Ali*et al.*, 2015) in Matthew, Mahamadou*et al.*,(2016).SDN can be seen as an eye-catchinghoneypot for intruders and a source of challenges for less equipped network operators such asamplifiedprospective for denial-of-service (DoS) attacks.OpenFlow is exposed to man-in-the-middle attacks when Transport Layer Security (TLS) is not used and network breaches may result when network controllersare shared by multiple users or applications(Ali *et al.*, 2015) in (Matthew, Mahamadou&Sarhan. 2016). Rapid changes in position and strength of flows requires flexible move toward successful network resource(s) management, various number of devices such as smartphones, tablets, and notebooks had increased much fold to put pressure on enterprise resources to bring about rapid changes to network resources and as such security challengesto the management of Quality of Service (QoS) (Muhammad *et al.*, 2014).

Internet with the use of traditional IP based protocol has exploited it functionality and there is a need for a network paradigm that will take the network to a new level,suitable for today'sdemand of internet and its functionality. Software Defined Network (SDN) promised potential basic change in network configuration and real-time traffic management performed (Taimur, 2017). It separates between the network control plane and the data plane, which provides user applications with a centralized view of the distributed network states (Ian *et al.*, 2016). It moves the control plane outside the switches and enables an external centralized control of data through a logical software entity known as the SDN controller., it decouples software from hardware and centralizes

network state in the control layer(Ian*et al.*, 2016).This makes the network administration, provisioning, arrangement, resource optimization, and network protection flexible using robotic SDN programs (Vandana, 2016).This enables researchers and practitioners to design much easier, flexible and powerful innovative network functions and protocolscalled SDN (Seungwon*et al.*, 2013). It enables direct programming of network operation(s) using an ordinary computer, programmer, operating system and programming languages.SDNs are logically segmented on three general regions: Application layerthis is the management plane responsible for the network programming section. Control layer hosting the network intelligent and Datalayer(Bruce & Rossi, 2016).

The remainder of this paper is organized as follows: Section 2 is the background of the work, Section 3 introduces the framework for preventing Software Defined Networks (SDN) from Malicious Attacks, Section 4 describes the result derived from the given framework in Section 3. Finally, an important conclusion is discussed in Section 5.

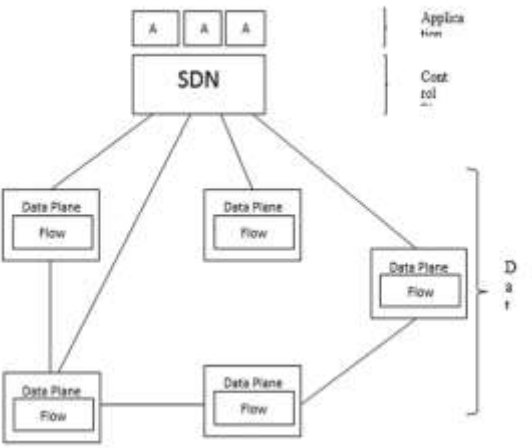

Figure 1:Overview of Software Defined Network (Sdn) (Okunade & Osunade, 2014)

2. Background of the Work

2.1. Northbound Application Programming Interfaces (APIs)

This is an open source-based application interface representing the software interface between the software modules of the controller platform and the SDN applications. The northbound interface facilitates the operation by providing the abstract view of the underlying network and empower the direct expression of network behavior and requirements.

2.2. Application Plane/Layer

Application plane is the topmost SDN planethat process request of incoming traffic and request services from the lower layers on behalf of the received traffic for further processing (Hrishikesh, 2015)is composed of network service applications, business

services, security services, and others that benefit from abstracted global view of the network according to their own purposes (Cabaj*et al.*,2014).this is an example of Northbound Application Programming Interfaces.

2.3. Control Plane

Control Plane handle the network intelligence control and states, it implement the network policies to globally regulate the network states and activities of the SDN. The logical centralization of controller enabled better decision making and maintaining of a global view of the entire network (Chienhung, Kuochen, and Guocin, 2017). It is the brain behind the successful execution of any SDN activities. According to Daojing, Sammy and Mohsen (2016), control plane manages the configuration of networking devices (such as switches and routers) and their forwarding functions. The data plane consists of protocols to execute the forwarding functions according to the rules configured by the control plane protocols. SDN controller is the central point of the network that enables the administrator to apply custom policies/protocols across the network hardware; control plane directs the data plane on flow forwarding and modifications processes. The controller is accountable for the conversion of applications' orders to the lower level communication protocol used by the data plane devices(Cabaj*et al.*,2014). The most widely deployed controller is a network operating system (NOX), controller. Naga*et al.* (2015) made to understand that controller can exercise it dynamic nature to modified the switchesthrough commands to adjust to traffic requests and equipment failures that may be observed through an event.

2.4. Southbound Application Programming Interfaces (APIs)

This is an interface through which the controllers are able to communicate with the network devices such as switches and data plane.It empowers the direct expression of network behavior and requirements. A controller can implement its responsibilities on data plane by communicating its command to the data plane through the southbound such as changing of forwarding behavior of a switch through altering offlow rule. Southbound Application Interface (APIs) are communication protocols between the controllers and the data planes examples are;OpenFlow (SDN most widely used communication protocol), OVSDB, OpenDaylight, Onix and HP VAN, and so on.

2.5. OpenFlow(OF)

OpenFlow communicate between the SDN controller via southbound open interfaces (such as OF protocol)and the data plane.OFwas created and hosted at the University of Stanford in 2008 for evangelizing and supporting the OpenFlow Community. OpenFlow is the most widely used SDN protocol; it is an open standard based communication protocol that enables the control plane to communicate with the data plane according to (Mateus, Bruno and Katia, 2013).Wolfgang and Michael (2014) stated that OpenFlow mainly focuses its consideration on switches whereas other SDN approaches focused on other network elements such as routers. According to Jad, David, Covington, Guido and Nick (2008) OpenFlow pushes difficulty to controller software so that the controller administrator has full control over it. This is done by pushing forwarding decisions to a "logically" centralized controller and allowing the controller to add and

remove forwarding entries in OpenFlow switches.

3. Algorithm and Implementation

3.1. Methodology

To address the aforementioned problem, a code was written to implement the Security Algorithm presented with embedded security extension of SDN OFtable rule (figure 2). This introduced security controlextending the SDN flow table with black/white list, which helped to secureSDN paradigm, where control plane will check for the authentication of users' application through the API foruser's confirmation usingwhite / blacklist for legitimacy confirmation of users' request who is requesting to make use of control plane by sending signals.

3.2. White / Black List plus Hash Bayes theorem (W/B+HBT) Algorithm Model

Figure 3 is the W/B+HBT Security Algorithm for preventing malicious attacks in Software Defined Network and process model that shows the incoming packet/request from the network, parsing the header field and match against the flow table to check if flow rule is already presented for the source address. If checked result is (NO) it means no existing flow rule for the packet source address, implying that packet/request source is communicating with that particular destination for the first time. The algorithm then requests from the controller for the creation of new flow rule for the newly arrived requestpacket transmitting from an unknown source. If the test checked result is (YES) it means there is an existing flow rule between the source and destination of the newly arrived packetrequesting from the network. The algorithm further its test to check if the identified flow rule between the basis and target of requesting packet is enlisted within the black or white list security extension of the SDN OF Architecture. If the flow rule is within the white list, the transaction is successfully executed by adding an entry for it in each of the switches along the path. Otherwise (if the flow rule falls within the blacklist), the algorithm generates an alarm that is sent to the controller and it also replies by sending a drop action to block or discard the transaction

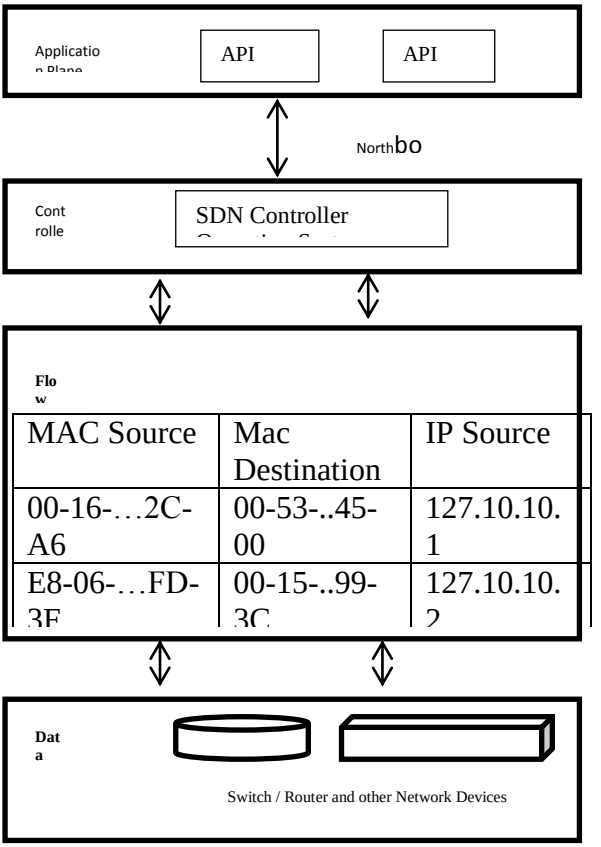

Figure 2:Extended SDN Openflow (Of) Table with White/Blacklist Security Features

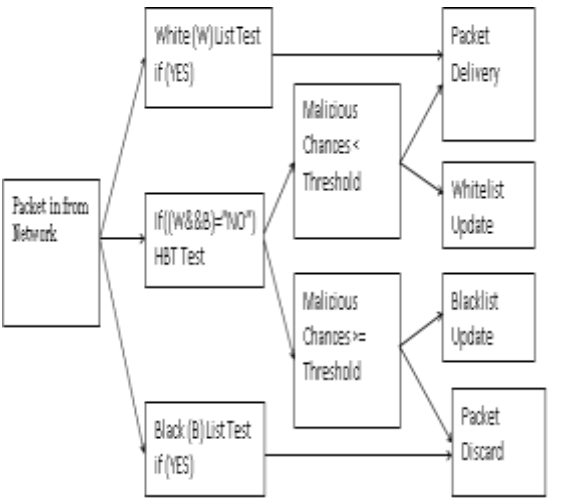

Figure 3: W/B+HBT Algorithm Model

If otherwise (newly arrived Packet source) not in either Black/White list, then the algorithm applies HBT content-based filter to calculate the Malicious chancesof the incoming packet using statistical Bayes' theorem (Okunade &Osunade, 2014).If (Malicious value) less than (<) the set Threshold of 0.5 the packet is forwarded/delivered to the appropriate quarters. If the packet calculated Malicious chance/value is greater than (>) the set Threshold of 0.5 the packet is discarded. Whatever the case may be the result is used to update the white/blacklist for a subsequent transaction(s).

S/NO	TOKEN	SPAMICITY
1	Kin	0.997472
2	Lottery	0.997382
3	Top	0.995529
4	Email	0.993583
5	Million	0.993062
6	Win	0.991283
7	Account	0.98367
8	Invoiced	0.983607
9	10.1.1.27	0.4
10	209.11.24.18	0.4
11	Abacha	0.643038
12	About	0.247848
13	After	0.19774
14	All	0.400717
15	And	0.5
16	Another	0.299899
17	App2.incamail.com	0.4
18	Are	0.404241

Figure 4: Some Suspicious Tokens and Associated Spamicity /Malicious Values

3.3. Word Hashing Operation

Word hashing operation is foremost executed on the newly arrived packet content, this is the removal of all unwanted prefixes, affixes and suffixes in the word(s) in order to deal with actual root/real word. The security algorithm contains inbuilt word hashing filtering technique that removed all unwanted prefixes, affixes and suffixes special characters used around the word(s) (especially around the suspicious terms) by intruders to misspelled/manipulate/ modified/mismanage tokens (such as $, /, \, |, =, !, @, #, %, ^, &, , (,), <, >, ?, :, ", ', {, [, },] and so on) used to foil the filters. This is done on the words in order to deal with actual root/real word, needed to calculate the malicious chances value using the Bayes' theorem.Then, algorithm will match packet token one after the other against suspicious table's token (Figure 4) in the database one after the other till the end of the suspicious table's token and then take the next token/word from the packet and do the same thing till the end of the tokens/words in the packet and match it against the list of tokens in the suspicious table.Then if there is matched the spamicity value of that particular matched token in the suspicious table (Figure 4) will be retrieved and assign against "a" been the first matched suspicious token follow by next matched identify suspicious term/token and assigned "b" been the second matched suspicious token, up to the last matched suspicious token and assigned it "z" is the last matched suspicious token. This assigned alphabet a to z are the alphabet finds in the Bayes formula, and spamicity values in (Figure 4) assigned to each of this alphabet (a-z) will be substituted into the Bayes formula as showed:

$$p(a,b,c...z)=\frac{a*b*c*.....*z}{a*b*c*.....*z+[(1-a)*(1-b)*(1-c)*.....*(1-z)]}$$

Then use the Bayes formula to calculate malicious chances, result gotten out of values substituted into the formula will then check against the threshold value that could set to any of: minimum with threshold value of 0.2, medium with threshold value of 0.3 and maximum with threshold value of 0.5 to give if (maliciousChances<= threshold) the entire newly arrived packet is forwarded to the appropriate port and then populate packet table of SDN database whitelist. But if otherwise (maliciousChances> threshold) the entire newly arrived packet is discarded and then populate a malicious table of SDN database blacklist.

4. Results and Discussion

This report the results of basic evaluation of a prototype implementation of Software Defined Network (SDN) Security Access control Algorithm using PhP/HTML code, Running/loading the Algorithm is depicted in figure 5 below.It shows that the contents of flow table consist of previous transactions status between nodes that could be used to predict further transaction, it contain source and destination of transactions nodes IP and MAC addresses, action(s) performed on such transaction which could either be "drop" or "forward to the appropriate quarters", security status that could be grouped into "blacklist" or "whitelist" and update status that signified if the flow table was initially populated at the starting point of implementation or updated by the application based on encountered during the execution of the application and also stated the date and time updated.

Figure 5:View Flowtable

Malicious Inbox (Figure 6) is the list of received malicious packets, these are the list of incoming packets that are classified to be malicious rather than been packet. They are an unwanted packet and identified to be dangerous.The experiment was able to successfully group the entire algorithm

tested malicious packets as such (malicious) therefore recorded 10% false positive and false negative.

Figure 6: Malicious Inbox

Packet Inbox in Figure 7 is the list of received legitimate packets, these are the list of incoming packets that are classified to be legitimate rather than been malicious. The experiment was able to successfully group the entire algorithm tested legitimate packets as such (legitimate) therefore recorded 90% true positive and negative.

Figure 7:Packet Inbox

4.1. Evaluation of Algorithm with the Existing TopoGuard Security Method

In an existing TopoGuard Security Method in Figure 7, once a packet send to an host could be hijacked, subsequent packets supplied to that particular host would be completely hijacked and redirected to the hijackers. The chart shown in figure 8 represents an evaluation of the White/Blacklist plus Hash Bayes Theorem (W/B + HBT) Algorithm with the Existing TopoGuard

Security where the two security methods were tested with the same data. The implemented TopoGuard Security algorithm indicates that 80% legitimate and malicious packets were classified as True positive (+ve) and true negative (-ve) where 20% legitimate and malicious packets were classified as False positive (+ve) and false negative (-ve). Whereas the White/Blacklist plus Hash Bayes Theorem (W/B + HBT) gives success record of 90% True positive (+ve) and true negative (-ve) and 10% record of False positive (+ve) and false negative (-ve) of legitimate and malicious packets classification.

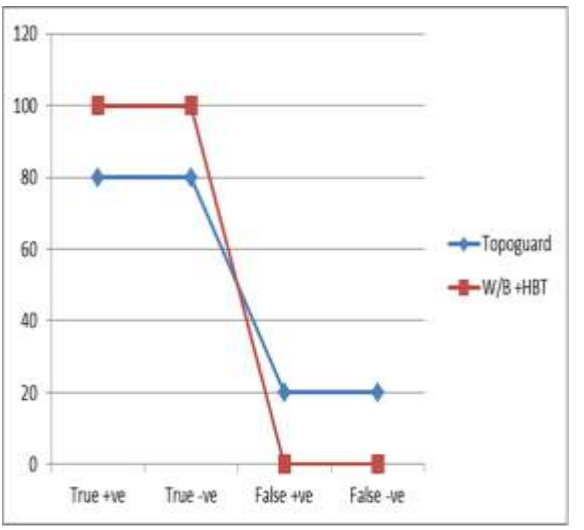

Figure 8: Evaluation of Algorithm with the Existing Topoguard Security Method

4.2. Discussion of Result

The presented algorithm having combined three examination levels. White/Blacklist plus Hash Bayes Theorem (W/B + HBT) Algorithm implementation prevented false positive or negative packets from being present. Unlike the existing Topoguard security method that discovers and prevents packet(s) from being sent to a changed or modified host address/location only, but does not prevent the host address/location from being changed or modified. The algorithm (W/B + HBT) prevents the insecure source from sending a packet to the targeted host and also prevents insecure (malicious packet(s)) from been sent. Whereas the existing Topoguard security method only considered already hijacked host (using source modification) from receiving the packet.

The result of evaluation of presented White/Blacklist plus Hash Bayes Theorem (W/B + HBT) security Algorithm compared against the existing Topoguard security Algorithm, recorded that the existing Topoguard security Algorithm has in its records 20% false positive and false negative and 80% true positive and true negative. Whereas the W/B + HBT Algorithmhave the result of 10% false positive and false negative and 90% true positive and true negative, which is accurate packets classification

and far better, compared with the existing Topoguard security Algorithm.

5. Conclusion and Recommendation

5.1. Conclusion

This paper discussed in details the developed algorithm that prevents Software Defined Network (SDN) from malicious attack. As a proof of concept, it has been demonstrated and concluded from findings that algorithm combined source identification/authentication (using white/blacklist) and content filtering (using word hashing and Bayes' theorem) (W/B + HBT) method of malicious identification/authentication and packet grouping, provides effective solution to legitimate/malicious mail identification/authentication and as such prevents malicious attack from accessing their targeted host in Software Defined Network. The experiment was a successful one recorded 10% false positive and false negative, and 90% true positive and true negative.

5.2. Recommendation

This paper recommends the use of combined methods of source identification using whitelist/blacklist combined with word hashing and Bayes' theorem for content filtering mechanisms/algorithm (W/B + HBT) as a preventive measure for intrusion prevention in Software Defined Network (SDN).

References

Ali, S. T.,Sivaraman, V., Radford, A., &Jha, S. J. (2015). A survey of securing networksusingsoftwaredefined networking. *IEEE Transactions on Reliability*, vol. 64, no. 3, pp. 1086– 1097.

Bruce, H. & Rossi, R. (2016). Software Defined Networking for Systems and NetworkAdministration Programs. *The USENIX Journal of Education in System Administration*. Volume 2, Number 1. *www.usenix.org/jesa/0201*

Cabaj, K., Wytrębowicz, J., Kukliński, S., Radziszewski, P. & Truong D. K. (2014). SDN.Architecture Impact on Network Security. Position papers of the Federated *Conference on Computer Science and Information Systems*, pp. 143–148. ACSIS, Vol.3

Chien-hung, L., Kuochen, W. &Guocin, D. (2017). A QoS-aware routing in SDN hybrid networks. *The12th International Conference on Future Networks and Communications*. Published by Elsevier B.V. Peer-review under the responsibility ofthe Conference Program Chairs. Procedia Computer, Science 110 pp. 242–249. Available online at www.sciencedirect.com

Daojing, H., Sammy, C.,& Mohsen, G. (2016).Securing Software Defined Wireless Networks. 0163-6804/16/. *IEEECommunications Magazine*.

Diego, K., Fernando, M. V. R. & Paulo, V. (2013). Towards Secure and Dependable Software-Defined Networks.HotSDN'13, Hong Kong, China. ACM 978-1-4503- 2178- 5/13/08

Hrishikesh, A. D. (2015). Software Defined Networks: Challenges, Opportunitiesand Trends.*International Journal of*

Science and Research (IJSR) Vol. 4 (9). www.ijsr.net Licensed Under Creative Commons Attribution CC BY

Ian, F. A., Ahyoung, L., Pu, W., Min, L. & Wu, C.(2016). Research Challenges for Traffic Engineering in SoftwareDefined Networks.*IEEE Network*. pp. 52-58

Jad, N., David, E., Covington, G. A., Guido, A. & Nick, M. (2008). Implementing an OpenFlow Switch on the NetFPGA Platform. *ANCS '08, San Jose, CA*, USA. ACM 978-1-60558-346-4/08/0011.

Mateus A. S. S., Bruno, A. A. N. & Katia, O. (2013). Software-Defined Networking Based Capacity Sharing in Hybrid Networks. *IEEE*. http://www.projectfloodlight.org/floo dlight/

Matthew, N. O. S., Mahamadou, T. &Sarhan, M. M. (2016).Software-Defined Networking Concepts.*Journal of Scientific and EngineeringResearch*Article 3(5): 92-94.

Muhammad, H. R., Shyamala, C. S., Ali, N. &Bill, R. (2014). A Comparison of Software Defined Network (SDN) Implementation. *2nd International Workshop on Survivable and V. Robust Optical Networks* (IWSRON). Procedia Computer Science 32, pp. 1050 – 1055.

Naga, K., Haoyu, Z., Michael, F. & Jennifer, R. (2015). Ravana: Controller Fault-Tolerance in Software-Defined Networking. SOSR. Santa Clara, CA, USA. *ACM* 978-1-4503-3451-8/15/06. *http://dx.doi.org/10.1145/2774993.2 774996*

Okunade, O. A. &Osunade, O. (2014). A Security Architecture for Software Defined Networks(SDN). *International Journal of Computer Science and Information Security*, Vol. 12 (12).

Raphael, H., Dietmar, N. & Mark, S. (2015). A Literature Review on Challenges and Effects of SoftwareDefined Networking. *Conference on ENTERpriseInformation Systems / International Conference on Project Management / Conference on Healthand Social Care Information Systems and Technologies*, CENTERIS / ProjMAN / HCist. ProcediaComputer Science 64 pp. 552 – 561. Available online at www.sciencedirect.com

Seungwon, S., Vinod, Y., Phillip, P. &Guofei, G. (2013). AVANT- GUARD: Scalable and Vigilant Switch Flow Management in Software-Defined Networks. CCS'13, Berlin, *Germany*. *ACM 978- 1-4503-2477 9/13/11.http://dx.doi.org/10.1145/25 08859.2516684.*

Systems -KES2013.Published by Elsevier B.V. Selection and peer-review under responsibility of KES International. Science Direct.22, pp. 810 – *819*

www.sciencedirect.com

Taimur, B. (2017). State of the Art and Recent Research Advances in Software Defined Networking. *Wireless Communications and Mobile Computing Hindawi.*Article ID 7191647, 35 pages https://doi.org/10.1155/2017/7191647

Vandana C.P. (2016). Security improvement in IoT based on Software Defined Networking (SDN). *International Journalof Science, Engineering and Technology Research (IJSETR*), Vol. 5(1). 292 ISSN: 2278 – 7798

Wenfeng, X., Yonggang, W., Chuan, H.F., Dusit, N. &Haiyong X. (2015).A Survey onSoftware-DefinedNetworking. *IEEE Communication Surveysand Tutorials*, Vol.17(1). pp. 27-51.http://www.ieee.org/publications_standards/publications/

Wolfgang, B. and Michael, M. (2014). Software-Defined Networking Using OpenFlow:Protocols, Applications and Architectural Design Choices. *Future Internet* 6, pp. 302-336; doi:10.3390/fi6020302 ISSN 1999- 5903. *www.mdpi.com/journal /future internet*

Yutaka, J., Hung-Hsuan, H. and Kyoji, K. (2013). Dynamic Isolation of Network Devices UsingOpenFlow for Keeping LAN Secure from Intra-LAN Attack. *17th International Conference in Knowledge-Based and Intelligent Information and Engineering Systems* - KES2013. Published by Elsevier B.V. pp. 810 – 819. ScienceDirect Available online at www.sciencedirect.com