Security Analysis as Software-defined Security for SDN Environment

Nadya EL MOUSSAID
ESSI, National School of Applied Sciences, Ibn Zohr University, Agadir, Morocco
Nadya.elmoussaid@edu.uiz.ac.ma

Ahmed TOUMANARI, Maryam EL AZHARI
ESSI, National School of Applied Sciences, Ibn Zohr University, Agadir, Morocco
atoumanari@yahoo.fr, maryam.ensa@gmail.com

Abstract— **The security of cloud environment is always a target for attackers in order to exploit any of the system's vulnerabilities. Recently, software-defined systems (SDS) has become a focus of several researches. Where, SDS is in the way to replace the traditional networking, in order to provide facilities which are based on remote and centralized control. The security of SDS is a major requirement to guarantee the integrity, confidentiality and availability of data and the communication. This paper presents a security analysis as a software-defined security service that enforces the security within the SDN in the cloud environment. The security analysis is specified through the attack graph and alert correlation clustering, which aims to enhance the work of other security approaches such as IDS by giving a global view and hint about the security state of the environment, also by reducing the rate of false positive alerts.**

Keywords- Software-defined network (SDN), Security, Cloud computing.

I. INTRODUCTION

Virtualized technologies are the backbone of Cloud computing. Cloud computing provides flexible and dynamic services, namely, (i) On-demand capabilities, (ii) Resource pooling, (iii) Broad network access, (iv) Rapid elasticity and (v) Measured services. However, due to attacks aiming to exploit vulnerabilities of cloud computing systems in order to benefit from its advantages, enterprises doubt the level of security guaranteed, by the provider of cloud computing, for their sensitive data. The confidentiality, privacy, integrity and availability issues may affect the performance of services hosted within the cloud environment [5]. These issues become more critical when it comes to applying SDS in the cloud environment. Therefore, the necessity of the security after the introduction of SDS to the cloud is a requirement [5].

Software defined system employ software-defined in order to virtualize and automate the components of the company's IT architecture, to be presented "as a service".

We divide SDS into five major components:

- Software-defined Networking (SDN)
- Software-defined Cloud Networking (SDCN)
- Software-defined Storage (SDS)
- Software-defined Data Center (SDDC)
- Software-defined Radio (SDR)

There is several benefits of using each component of SDS, but they all provide the major five benefits [6], that we quote:

Accuracy: The use of software-defined technologies makes the IT resources automated and programmable. Also, render the client's request independent of the hardware that the company runs. SDS can reduce the rate of errors committed by human interactions or triggered by machines, where, the components are more intelligent and precise.

Agility: agility is one of the cloud characteristics as well as SDS, where, the ability of moving from a system to another or configuring by adding, deleting or migrating components from an environment to another is an easy and flexible task.

Adaptability: the network or data storage are based software-defined system enhances the no reliance on hardware resources of vendors, which leads to a flexible way to manage changes and the ability to adapt in a new environment.

Assurance: Each enterprise has a specific policy to build. SDS provides more insurance to organizations in regards to the infrastructure that share same standards and configurations.

Software-defined networking (SDN) [1,5] represents a novel network architecture, which is based on network function virtualized (NFV) [7,8] that consists of several virtual machines (VM) running processes and softwares. OpenFlow was the result of collaboration between Stanford University, Princeton University and other universities researchers [2], it is the main factor that enables SDN to communicate with its components [1].

Many researches focus on how to ameliorate the controller design, the optimization of routing decision, the performance of forwarding, or saving the energy of a datacenter, etc... [4] Without taking in consideration the requirement of security policies within SDN. Therefore, the security of SDN is a huge challenge since SDN is a novel domain of researches [3].

In this paper we are concentrating on how to analyze the security state of cloud environment through security analysis as a software-defined security in order to guarantee the security properties namely confidentiality, integrity and availability. For that purpose, we have opted to use the attack graph technique in collaboration with alert correlation clustering to decrease the rate of false positive alerts.

The rest of the paper is structured as follow: Section II gives a brief overview about the architecture of SDN and its

components, then Section III presents the different attacks against SDN. Section IV presents some related works. Section V describes our approach security analysis as software-defined security, and we will summarize this paper by results of our experiments in section VI and a conclusion in section VII.

II. Overview of SDN architecture

SDN characterized by the fact it is centralized on application level and distributed on infrastructure level. Therefore, the global network is managed by a component called "controller". The controller manages the subset of components through OpenFlow. Fig. 1 presents the basic structure of SDN that contains three major layers which is Application layer, Control layer and Infrastructure layer, where, control plane is separated from data plane.

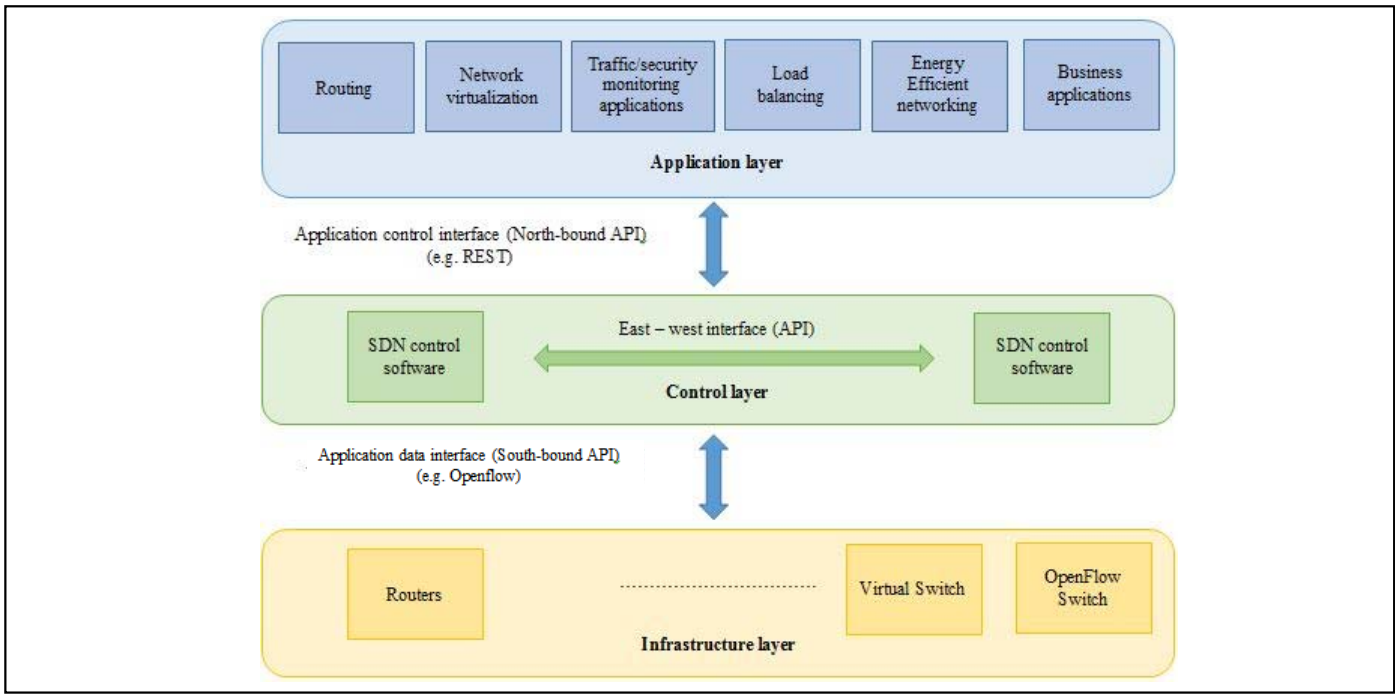

Figure 1: Architecture of SDN

A. Infrastructure layer

Infrastructure layer or data forwarding layer consists of several network devices namely SDN switches, routers, etc... Each switch contains a flow table responsible of forwarding packets through rules [9 - 11]. Table flow entries consist of three parts such as the pattern, action and stats [6]. The pattern is the header field of a packet, and the action is performed according to the match of a set rules. Stats are a set of indication that indicates the status of network. These indications can be counters, priority or a cookie [12].

B. Control layer

Control layer is the core of an SDN, which consists of controllers that take responsibility of managing traffic flow in a network, flow forwarding and taking decision on routing using programming. Controllers can communicate with each other through east - west interfaces in order to maintain the network connectivity and status [7]. And through north-bound APIs (e.g REST, frantic, etc.) to application layer, and through south-bound APIs (e.g. OpenFlow, NetConf, etc) to infrastructure layer [7-9].

C. Application layer

Application layer is the layer built on top of the SDN controller; it contains an ensemble of applications related to network virtualization [13], traffic/security monitoring application [12], Intrusion detection systems, prevention detection systems [12], load balancing [14], business application, etc...

III. SDN security issues and attacks

A. Security Issues

SDN technologies are widely used in different areas such as cloud computing, data centers, wireless networks, cellular networks, etc., in order to benefit from the advantages offered by SDN. Therefore the security is highly recommended. Thus, in this section we present the most important security issues and attacks faced by SDN. Traditional attacks remain in SDN technology and reduce its efficiency and its performance.

The architecture of SDN provides several benefits and advantages to facilitate the tasks. However, SDN suffer from some security issues, we quote:

Open Programmable APIs can cause security issues, where, attackers can exploit the openness nature of these APIs to several layers. Either, it makes vulnerabilities of software more visible to attackers and may lead to inject malicious code or exercise any attack such as Cross-site Scripting (XSS) [15-16].

Controllers issues, even the benefits of using a central architecture of SDN to configure the network, fast and efficient routing calculation and reducing the complexity of the network, it may be an easy target for attackers. An unauthorized penetration to the SDN controller may lead the attacker to take the whole of a network which causes a massive damage [12].

SDN switches issues focuse on the vulnerability of Flow table entries limitation which makes it very sensitive to DDoS attacks.

B. Attacks faced by SDN

Attacks against Software-defined networking are diverse according to the layer and level where it is exercised such as application layer, Application & controller interface, controller layer, controller & data interface and Infrastructure layer (or data layer).

1) Application layer attacks:

Unauthorized Access: The application layer is built on SDN control layer, which contains the concrete part of SDN functionalities in term of applications, where, the control layer is abstract part of network physical resource. In case of an attack, attackers try to gain unauthorized access to sensitive data, the information and configuration of the network without Authenticating application within SDN environment. Most of applications are developed by third party not by controller vendors, that doesn't take into consideration the security requirements during the development phase [12], and make applications suffer from various security issues [15].

Rules insertion: SDN is widely used in different domains such as cloud computing with a huge number of complex applications and services. Therefore, creating and inserting security rules may lead to conflicts.

Malicious code and programs injection: According to OWASP project [15], injection in its different forms is one of top ten attacks, where, attackers inject malicious code/ scripts/ programs or flaws which may lead to information corruption or loss, etc...

2) Control layer Attacks

Control layer is the main component of SDN that takes control of the network. Therefore, because of its importance, an attack can lead to serious problems and disability of the whole network such as unavailability. In this section, we quote well known attacks against control layer.

Attacks from application: As it's mentioned previously, the application layer is built on control layer. Hence, An attacker that gains unauthorized access from the application layer can get information and sensitive data about the network configuration that may lead to another attack against control layer.

DDoS/ DoS attack: Denial of service and distributed DoS is one of the common attacks and threats exercised by attackers. DDoS/ DoS is a challenge for SDN controllers that influences the availability of the network to legitimate users [12][17]. DDoS/ Dos attack can be exercised through several ways and levels (i.e. at controller, switchers or at the channel between controller and switchers), where, attackers aim to flood the network by sending flooded requests that consume the resources of a controller and put it in a saturated situation [18].

Attacks against distributed multi-controllers: Due to the increasing of network range, distributed multi-controller come to divide the network into sub-networks for a better management. However, this solution may lead to some security issues such as establishing a performing security policy and managing conflict of configurations. Also, one of the purposes of distributed multi-controllers within control layer is to solve the problem of DDoS / DoS attacks and decrease the damage of these attacks. However, distributed controllers remain vulnerable to it [12][19].

3) Infrastructure layer attacks:

The OpenFlow switch contains three parts namely: OpenFlow client, Flow buffer and Flow table. Each part is a target of attacks.

DoS attack: The OpenFlow switch faces two main challenges which are: 1/ a limited number of entrees that Flow buffer supports and 2/ a limited number of flow tables. Once a packet is received, it's stocked in buffer flow till the insertion of a new rule in the case of a new packet, or till the response of a rule in the flow table. Therefore, an attacker can saturate the buffer flow and the flow table by sending numerous large unknown packets, which will generate new rules to be inserted into flow tables. This attack leads to compromise the forwarding of legitimate packets by dropping them [19][20].

Man-in-the-middle: The attacker of man-in-the-middle (MITM) monitors the traffic between controllers and switches, in order to intercept the information of communication without being detected. Controllers and switches are not directly connected to each other, which makes each entity doubtful to be a MITM node [21]. MITM attack leads to the implementation of other attacks such us eavesdropping and black-hole attack [12][20].

IV. RELATED WORK

Detecting vulnerabilities of a system become the spotlight nowadays through attack graphs. Attack graph is a network vulnerabilities analyzer, which presents a different path that an attacker can use to gain access to unauthorized environment. The generation of attack graph phase contains three steps: Attack graph modeling, Information collection and Attack graph core building [22]. Attack graph modeling is a set of nodes and edges, where, nodes represent the exploit of vulnerabilities, and edges represent the correlation between two vulnerabilities or an attack transition. Attack graph modeling is based on the network topology and the running applications within the network hosts and their relationships [22] [25].

Several tools are proposed by researches to generate attack graph, we quote: MulVAL (Multihost, multistage Vulnerability Analysis) is based on datalog as its modeling language [23][25] to generate an attack graph. Authors of [24][25] present an attack graph modeling based on JIGSAW, which is an attack specification language. TVA (Topological

Vulnerability Analysis) is an attack graph tool that analyzes the network vulnerabilities automatically [26][25]. NetSPA (A Network Security Planning Architecture) is a network attack graph tool, that identifies attack path from the network topology [25][27].

The alerts are generated by different sensors such as IDS/IPS, firewalls or snort. And because of the huge number of attacks against the cloud and their complexity, the number of alerts increases [30][31]. The alert correlation framework may consist of: Normalization, Aggregation, Correlation, False Alert Reduction, Attack Strategy Analysis and Prioritization [31]. In [31], authors have proposed an alert correlation algorithm based on AG that helps to detect attack's scenarios. Authors of [30] authors have proposed two novel alert correlation approaches based on policy enforcement and defender capability models and information security indicators.

V. SECURITY ANALYZING AS SOFTWARE-DEFINED SECURITY

In this section we give a description of different components of our model.

1) Model's componenents

Security analysis as SDS contains the following components as it is mentioned in Figure.2.

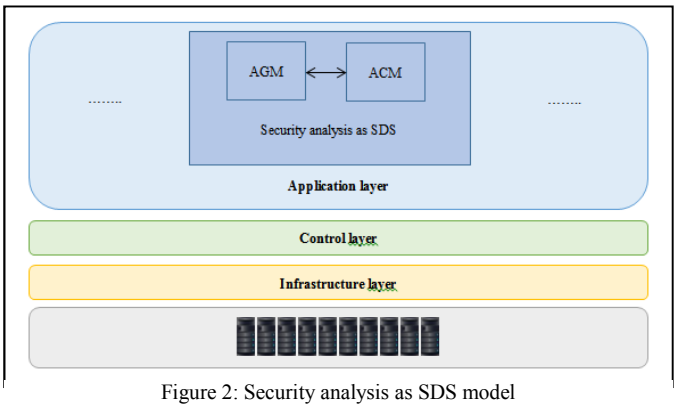

Figure 2: Security analysis as SDS model

- **Attack graph model**: the attack graph is responsible of measuring the ability of the environment to overcome attacks by providing different paths that an attacker can take through exploiting vulnerability of the environment.
- **Alert correlation model**: This model comes to solve the problem of high rate of false positive. Also gives a description of the detected attack.

2) Attack graph model (AGM):

Attack graph is a tool that describes the state of network; also it shows how an attacker can exploit the vulnerabilities of the network. Therefore, it aid to identify potential attacks through identifying the vulnerabilities within cloud environment. An attack graph should take into consideration multi-stage and multi-host attack paths [23].

As mentioned previously an attack graph is a set of nodes noted "N" and directed edges noted "E".

Definition 2.1: $N = \{n_1, n_2, n_3, \dots\dots, n_n\}$ is a set of nodes. A node can be a new exploit or a result of a previous exploit by the attacker.

Each node is defined by (host, CVE_ID, programme), a host can be a virtual machine (VM), switch or controller. Example: (Open vSwitch_ID, CVE-2016-2074, ovs-vswitchd) this vulnerability leads to buffer overflow in lib/flow.c of ovs-vswitchd.

Definition 2.2: $E = \{ e_1, e_2, e_3, \dots, e_n\}$ is a set of directed edges.

The attack graph is based on the security metrics mentioned in the following section.

3) Security metrics:

Security metrics represent the characteristics of vulnerability, which measure the condition of a network security under the current configuration. For that purpose, we have used the Common Vulnerability Scoring System (CVSS) [28] that provides standardized vulnerability scores and helps prioritize risk. CVSS contains three major vulnerability scores namely: Base score (BS), Temporal score (TS) and Environment score (ES).

Base score (BS): BS is a set of metrics that represent the characteristics of vulnerability that are constant over the time and environment.

Temporal score (TS): TS measures the current state of exploited techniques or code availability, which changes over time.

Environment score (ES): These metrics enable the analyst to customize the CVSS score depending on the importance of the affected IT asset to a user's organization, measured in terms of complimentary/alternated security controls in place, confidentiality, integrity, and availability. The metrics are modified equivalent of base metrics and are assigned metrics value based on the component placement in organization infrastructure [28].

Each score contains other attributes for a good description of vulnerability characteristics. In our model we have used the environment score (ES) to analysis the risk of the environment. And the main reason of using ES is that BS and TS are given by vendors, while ES are given by the end-user organizations because they are the only one who can estimate the impact of vulnerability within their computing environment.

For that we need to calculate the following metrics:

$$ES = 1.08 * (I + E) * E_{temporal} * RL * RC \quad (1)$$

Where:

RL is remediation level and RC is report confident.

The impact (I) is calculated through the impact metrics which are confidentiality impact (I_{Conf}), integrity impact (I_{Integ}) and the availability impact (I_{Avail}).

$$I = 7.52 * (ISC - 0.029) + 3.25 * (ISC - 0.02)^{15} \quad (2)$$

Where:

$$ISC = 1 - (1 - I_{Conf} * CR) * (1 - I_{Integ} * IR) * (1 - I_{Avail} * AR) \quad (3)$$

And the exploitability (E) is a product of attack vector (AV), attack complexity (AC), privilege required (PR) and user interaction (UI).

$$E = 8.22 * AV * AC * PR * UI \quad (4)$$

The value of I is between 0 and 0.915 and the ES value is between 0 and 10.

In order to measure the risk of a vulnerability to be exploited by an attacker, we calculate the initial probability values of vulnerabilities by dividing the ES by 10 [29].

$$P(n = True) = {ES}/{10} \quad (5)$$

Then we calculate the conditional probability of nodes according to their relation topology and the probability of their parents. Conditional probability measures the probability of attackers reaching their goal (exercising an attack).

$$P(n \mid parent(n)) = P(n) * \prod P(parent(n)) \quad (6)$$

4) Alert correlation clustering

Concerning the alert correlation, we have referred to the approach mentioned in [31]. For ameliorating the correlation, we have used on clustering method to classify alerts according to their resemblance and relation. An alert is an object that contains data related to the attack such as source address, destination address, type of alert and the time of the occurrence. Each alert is mapped to specific nodes in the AG using $map(alert)$ function [31]. After mapping the alerts to nodes we calculate the Euclidean distance between the object data (ES, Src, Dst, Time) and the center of the cluster. Therefore, similar alert are grouped into same cluster, which reduce the number of positives rate (Algorithm. 1).

Algorithm 1: Alert correlation clustering

Input: Alert, AG.
Output: Alert correlation clustering.
Start:
If(Alert is a new alert) **Then**
  n ← $map(alert)$
  ES ← $n.ES$
  Src ← $Alert.src$
  Dst ← $Alert.dst$
  Time ← $Alert.time$
  distance(ES, Src, Dst, Time, C)
  add(Alert, C)
End If
Update cluster
Return cluster
End

5) Security state

The security state is defined according to the value of the calculated probability.

Algorithm 2: Security state

If (probability is between 0 and 0.39) **Then**
  Security_state ← "***High***".
Else If (probability is between 0.4 and 0.69) **Then**
  Security_state ← "***Medium***".
Else Security_state ← "***Low***".
End If.

According to the algorithm 2, the security state (SS) can be categorized into three types namely: High, medium and low.The selection of countermeasures is based on the security state and the collaboration with Service Level Agreement (SLA).

VI. EXPERIMENTS

This section presents the results of the experiments that we have conducted on our work; we have created a virtual network using SDN. This virtual network contains 500 virtual machines with the following vulnerabilities mentioned in the table below:

TABLE 1: LIST OF VULNERABILITIES

Host type	Vulnerability	CVE-ID	Initial probability
Open vSwitch	Buffer overflow in ovs-vswitchd	CVE-2016-2074	0.83
Web server	Execute CodeOverflow	CVE-2016-1327	1
Virtual machine	The image build process	CVE-2016-4474	0.65
	Teradata virtual machine	CVE-2016-7488	0.78
Mail Server	Execute Code	CVE-2016-9920	0.88
	Denial Of ServiceCSRF	CVE-2016-4069	0.42
Servers	OverflowGain privileges	CVE-2016-1340	0.91
	The auth_password function in OpenSSH	CVE-2016-6515	0.62

For the purpose to test our work, we have performed a DDoS attack and multistep attacks against multiple servers and virtual machine. Then we generate a set of alerts to be used in alert correlation clustering.

The Alert correlation clustering (ACC) reduces the false positive alerts by 64% compared to the original false positive alerts that are generated by sensors (IDS…). Attack graph is created using the calculated ES. Therefore, the level of security state is defined through the calculated probability.

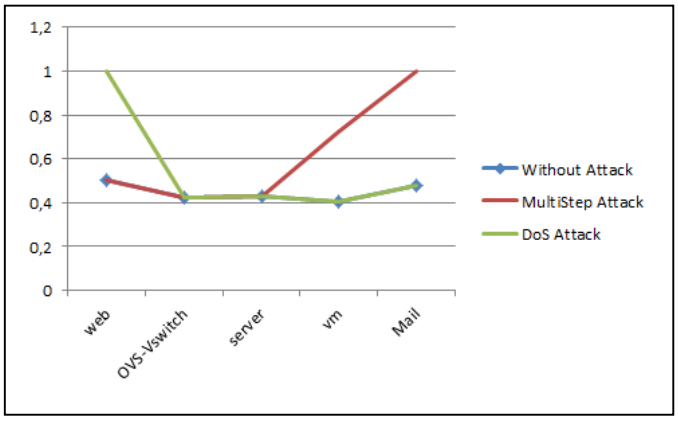

Figure 3: Security State

Figure.3 shows the changes of environment in the security state, which contains a set of applications. These changes are done by exploiting the vulnerabilities of the environment by attackers. In case of a successful attack, the probability equals 1 and the related probabilities are calculated.

To evaluate the system performance, we calculated the system's CPU consumption by sending a different amount of packets per seconds (Figure.4).

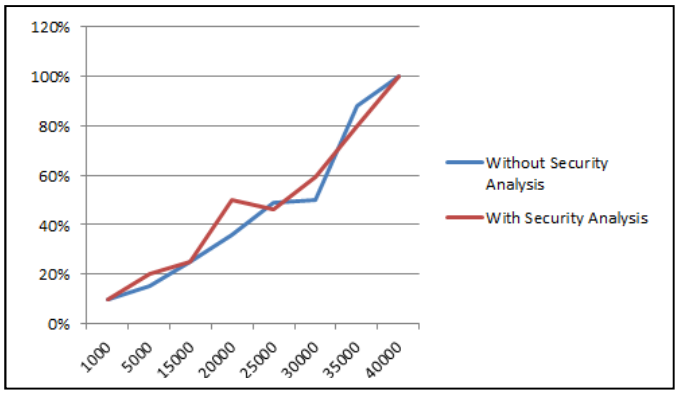

Figure 4: System's CPU consumption

VII. Conclusion

In this paper, we presented security analysis as a software-defined security to analyze the state of the environment by combining attack graph and alert correlation clustering. The security state is defined by calculated probability from environment metrics, which take into consideration the impact of an attack on confidentiality, integrity and availability. The further work will be aiming to propose the appropriate countermeasure algorithm.

References

[1] Cisco Inc, “Software-defined networking: why we like it and how we are building on it”, White Paper, 2013.

[2] N . McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson , J. Rexford, J. Turner, “OpenFlow: enabling innovation in campus networks”. ACM SIGCOMM Comput Commun Review, pp.69– 74, 2008.

[3] I. Ahmad, S. Nama,M. Ylianttila, A. Gurtov, “Security in Software Defined Networks: A Survey”, IEEE Commun Surv Tutorials, pp. 2317–2346, 2015.

[4] D. B. Rawat; S. R. Reddy, "Software Defined Networking Architecture, Security and Energy Efficiency: A Survey", IEEE Communications Surveys & Tutorials , pp. 1-22, 2016.

[5] F. Kemmer, C. Reich, M. Knahl, Nathan Clarke, "Software Defined Privacy", IEEE International Conference on Cloud Engineering Workshop, pp. 25-29 , 2016.

[6] Y. Gong, W. Huang, W. Wang, Y. Lei, "A survey on software defined networking and its applications", Frontiers of Computer Science, pp 827–845, 2015.

[7] B. Han, V. Gopalakrishnan, L. S. Ji, and S. J. Lee, “Network Function Virtualization: Challenges and Opportunities for Innovations”, IEEE Communications Magazine, pp. 90–97, Feb. 2015.

[8] W. Yang, C. Fung, “A Survey on Security in Network Functions Virtualization”, IEEE NetSoft Conference and Workshops (NetSoft), pp. 15-19, 2016.

[9] F. Hu, Q. Hao, and K. Bao, “A survey on software-defined network and OpenFlow: from concept to implementation”, IEEE Communications Surveys & Tutorials, vol. 16, no. 4, pp. 2181–2206, 2014.

[10] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, S. Uhlig, “Software-defined networking: A omprehensive survey”, Proceedings of IEEE, pp. 14–76, 2015.

[11] W. Stallings, “Software-Defined Networks and OpenFlow”, The Internet Protocol Journal, vol. 16, no. 1, 2015.

[12] Z. Shu, J. Wan, D. Li, J. Lin, A. V. Vasilakos, M. Imran, "Security in Software-Defined Networking: Threats and Countermeasures", Mobile Networks and Applications, pp. 764–776, 2016.

[13] D. V. Bernardo, "Software-defined networking and network function virtualization security architecture", https://tools.ietf.org/html/draft-bernardo-sec-arch-sdnnvf-architecture-00, 2017.

[14] S. Namal,I. Ahmad,A. Gurtov,M. Ylianttila, "SDN based intertechnology load balancing leveraged by flow admission control", IEEE SDN for Future Networks and Services, pp. 1–5, 2013.

[15] Top ten web application vulnerabilities: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project, 2017.

[16] M. Green, M. Smith, "Developers are Not the Enemy!: The Need for Usable Security APIs", IEEE Security & Privacy, pp 40-46, 2016.

[17] P. Zhang, H. Wang, C. Hu, and C. Lin, "On Denial of Service Attacks in Software Defined Networks", IEEE Network, 28-33, 2016.

[18] H. Wang, L. Xu, and G. Gu, “FloodGuard: A DoS Attack Prevention Extension in Software-Defned Networks,” the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 239-250, 2015.

[19] S. Scott Hayward, "Design and deployment of secure, robust, and resilient SDN Controllers",1st IEEE Conference on Network Softwarization (NetSoft), pp 1–5, 2015.

[20] K. Benton, L. J. Camp, C. Small, “OpenFlow vulnerability assessment”, 2nd ACM SIGCOMM workshop on Hot topics in software defined networking, pp. 151–152, 2013.

[21] St. B. Brezetz, G. B. Kamga, M. N. Balla, T. Criton, H. Jebalia, "SDN-based Trusted Path in a Multi-domain Network", EEE International Conference on Cloud Engineering Workshop, pp. 19-24, 2016.

[22] K. Kaynar; F. Sivrikaya, "Distributed Attack Graph Generation", IEEE Transactions on Dependable and Secure Computing, Pp. 519-532, 2016.

[23] X. Ou, S. Govindavajhala, A. W. Appel, "MulVAL: a logic-based network security analyzer", 14th conference on USENIX Security Symposium, 2005.

[24] S. J. Templeton, K. Levitt, "A requires/provides model for computer attacks", workshop on New security paradigms, pp.31-38, 2000.

[25] S. Yi, Y. Peng, Q. Xiong, T. Wang, Z. Dai, H. Gao, J. Xu, J. Wang, L. Xu,"Overview on attack graph generation and visualization technology", IEEE International Conference on Anti-Counterfeiting, Security and Identification, pp.1-6 , 2013.

[26] S. Jajodia, S. Noel, B. O’Berry, "Topological analysis of network attack vulnerability", the 2nd ACM symposium on Information, computer and communications security, 2007.

[27] K. Ingols, R. Lippmann, K. Piwowarski, "Practical Attack Graph Generation for Network Defense", 22nd Annual Computer Security Applications Conference, pp.121-130,2006.

[28] The Common Vulnerability Scoring System (CVSS): https://nvd.nist.gov/CVSS/, 2017.

[29] M. Frigault, A. Singhal, S. Jajodia, "Measuring Network Security Using Dynamic Bayesian Network", 4th ACM workshop on Quality of protection, pp. 23-30 , 2008.

[30] G. G. Granadillo, M. El-Barbori, H. Debar, "New Types of Alert Correlation for Security Information and Event Management Systems", 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp.1-7, 2016.

[31] S. Roschke, F. Cheng, C. Meinel, "A New Alert Correlation Algorithm Based on Attack", 4th International Conference on Computational Intelligence in Security for Information Systems, pp. 58-67, 2011.