Detection MITM Attack in Multi-SDN Controller

Anass Sebbar*, Mohammed Boulmalf†, Mohamed Dafir ECH-CHERIF El KETTANI*, Youssef BADDI‡

† ELIT-Internationale University of Rabat
Mohammed.boulmalf@uir.ac.ma
anass.sebbar@uir.ac.ma

ENSIAS-Mohammed V Rabat University,
Dafir@um5s.net.ma
anass.sebbar@um5s.net.ma

‡Université Chouaib Doukkali
Baddi.y@ucd.ac.ma

***Abstract*—Software Defined Security Networking is a new method to reduce risks while ensuring maximum protection for users and networks. It offers automated and usable intelligence to preserve the security of the systems. However, we have a variety of threats targeted to the plane and interface of SDN. In this paper, we present different attacks in Software Defined Networking (SDN) layers and interfaces, proposing two scenarios in order to describe the methodology of Man In The Middle (MITM) attack in different controllers like OpenDayLight (ODL), Open Network Operating System (ONOS) and RYU. We focus on the ODL controller which is the subject of this study. Commonly investigated types of vulnerabilities on SDN controller. The simulation results indicate that the attackers can control easily the SDN Controller, and communication between control layer and infrastructure layer is not secure. This result shows that ODL is vulnerable with respect to MITM attack. In this research, many recommendation and solutions measure to prevent and detect MITM attack is presented.**

***Index Terms*—SDN, Controller, ODL, ONOS, Ryu, Mininet, MITM**

I. Introduction

SDN has many terms such as Network Functions Virtualization (NFV), expects to change the way organize administrators manage networks by developing standard IT virtualization innovation to combine many system equipment sorts onto industry standard high volume servers, switches, and storage, which could be situated in Datacenters [20]. Software-Defined Wide Area Network (SD-WAN) is a particular utilization of software-defined networking (SDN) technology connected to WAN connections, which are utilized to interface undertaking systems - including branch workplaces and data centers - over vast geographic separations [1].

Software Defined Security Networking is a new method to reduce risks while ensuring maximum protection for users and networks. It offers automated and usable intelligence to preserve the security of the systems. However, we have variety of threats targeted to the plane and interface of SDN.

In order to secure SDN, it can give up a hardware and used only software applications for good protection must ensure that software application is not vulnerable.

This paper is intended to show the multiple vulnerabilities of SDN and how to ensure security in a controller to avoid attacks. The SDN has several advantages, and for many vulnerabilities targeting layers and interfaces of SDN. The majority of risk come from inside of the company, generated by different attacks, like DDOS, MITM and others, they can affect: Confidentiality, Integrity, Availability and Traceability of the SDN controller.

Our study focuses on how an attacker can use these vulnerabilities between Control layer and Infrastructure layer such as link connection, TCP communication, TLS, SSL, In order to investigate them, we created a simple Test-Bed.

The remaining of the paper is organized as follows. We discuss SDN definitions and terms in section II. We present our methodology in section III and our Results in section IV. Section V present conclusion and future work.

II. Background

A. Controler Overview

SDN draws attention as a network trend technology, has also several vulnerabilities. SDN controls the flow network through the controller. However, the controller can cause security problems. Indeed, the controller cannot correctly indicate a level of data if it infected by a malicious code or if it received a DDoS attack. Security of the controller is very important that controls the SDN network .

SDN is built around the Open API, mainly OpenFlow. It is one of the interfaces most commonly used to control the data layer. For this reason, architecture with enhanced security at the level of the operating system that controls OpenFlow is essential, and it is the starting point for the security in whole environment SDN.

Figure 1 shows a clear trend of three layers and two interfaces (Northbound & Southbound) of SDN controllers.

In the following we define the three controllers and the Mininet emulator:

Opendaylight (ODL): is an open modular platform to customize and automate networks of all sizes and scales. It is characterized not only by the implementation of OpenFlow, but also by implementing the startup controller SDN that integrates SDN protocols and other network functions [4].

Ryu: is a network defined by a component framework. It provides software components with a well-defined API that make it easy to developers to create new applications management and network control. Ryu supports various protocols for the management of network devices such as OpenFlow, Netconf, OF-config ... [5].

978-1-5386-4385-3/18/$31.00 ©2018 IEEE

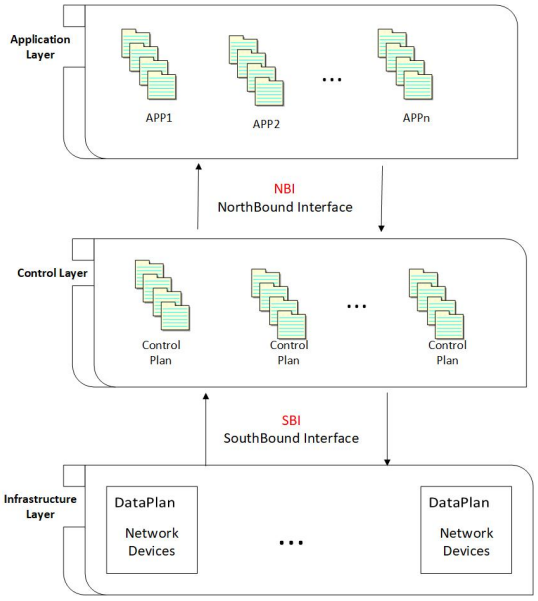
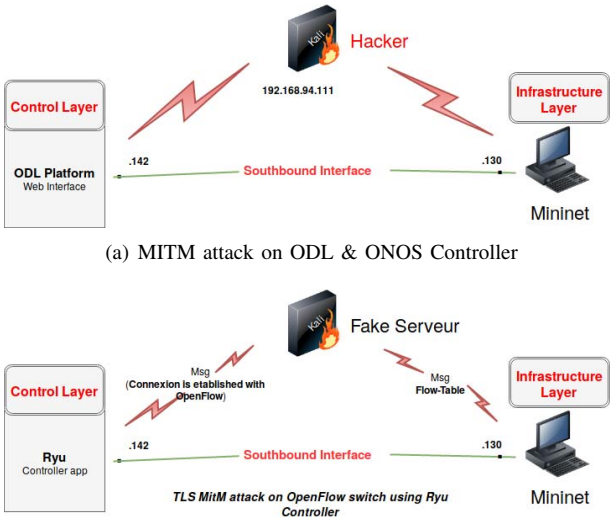


Fig. 1. SDN Architecture



(a) MITM attack on ODL & ONOS Controller

(b) MITM attack on Ryu Controller

Fig. 2. Vulnerabilities scenarios MITM attack

Open Network Operating System (ONOS): provides the control plan for SDN , managing components such as switches, network and links, and running software or modules to provide communication services to the final hosts and nearby networks [13].

Mininet Emulator: is an emulator that simulates a set of hosts, switches, network routers and create simple topologies. Mininet creates a devices like hosts that behave exactly like real machine [18].

B. SDN SECURITY VULNERABILITIES

We have variety of threats targeted to the plane and interface of SDN.

Table ref tab summarize different potential security issue and show in which SDN surfaces located as well to show the definition of attacks and these security services. We define the SDN security threats: **Unauthorized/Unauthenticated applications:** is the Act of accessing a network, system, application or other without authorization. On SDN network this attack occurs at the level of the application layer.

Fraudulent rule insertion affects the level of the northbound interface, is it the act of inserting malicious rules in the application layer and by that destroying the security of the network.

DOS or DDOS has a purpose to saturate or bomb a server, service IT and network. In the SDN network this attack will proceed at the level of the controller this mechanism can stop the controller to operate. Also **Controller Hijaking** attacks and **Unauthorized Access controller** target the part of the controller confidentiality [22].

Man in the middle attack occurs at the level of the SouthBound interface, where an attacker can intercept communications between controller layer and infrastructure layer without both parties.

Flow Table flooding attack routers or OpenFlow switches in the data plan, targeting the principle of availability or the security service [12], [19].

III. METHODOLOGIES

The MITM attack involves three protagonists: Control layer, Infrastructure layer, and Attacker. The figure 2 illustrate the goal who pretend to be the control layer or the infrastructure layer depending on how to communicate first, for example, if the control layer starts the communication, the hacker play the role of infrastructure layer. Which allows him to monitor all network traffic between layers, and to obtain information (passwords, access system ...) using ARP spoofing techniques. This technique allows to listen and corrupt but also to spoof an IP address or block traffic. This is useful for someone who wants to trick to be someone else on a network, the most common cases are Ethernet and wifi networks. [7] We have two senario:

senario 1: Figure 2-a shows that communication happens between the control (ONOS or ODL) and infrastructure (Mininet) through the southbound interface (SBI) layer. A Kali-linux machine trying to infiltrate inside the network. Kali-linux has infiltrated to the SBI interface with the arp spoofing technique it can send and receive information between the two sides here Kali-linux plays an attack role. To execute the MITM attack some tools are necessary such as ettercap (arp-spoofing) to sniffs live connections and TCP traffic with filtering content. We use wireshark to sniffs login and passeword and other interesting information aboute openflow protocole that passes through incrypted. Finaly we use SSLstrip to removes S in https making it unsecure, so we can follow disclosure of forward with both parties [9], [14].

senario 2: Figure 2-b shows that the Ryu controller communicate with the emulator Mininet through the SBI interface. To

TABLE I
SDN SECURITY THREATS & ATTACK SURFACES

References	Potential Security issue	Attack Surface	Definition of Attack	Attack Security Service
[18], [25]	Traffic flow Link	Infrastructure layer	Flow table Flooding	AVAILABILITY
[6], [17]	Forwarding and control plane communication	Southbound Interface	Man in the middle	Confidentiality & integrity
[27]	SDN Controller	Control layer	DOS, DDOS, Controller Hijacking, Unauthorized Controller Access	Confidentiality
[5], [8]	Application and Control layer Interface	Northbound Interface	Fraudulent Rule Insertion	Integrity
[13]	Malicious Applications	Application Layer	Unauthorized/Unauthenticated Applications	Confidentiality & Integrity

succeed the attack **TLS MITM** we use OpenFlow switch using Ryu controller [11]. First we generate an SSL certification, and then we launch Mininet emulator with a simple topology. In a terminal where we connect the controller Ryu to the Mininet remotely. Secondly, a message from the malicious application appear according to which application communication used, while running the attack the malicious application is used in a machine of Mininet a connection between the controller and the legitimate server is redirected to the fake server since our malicious application needs to connect to the fake server

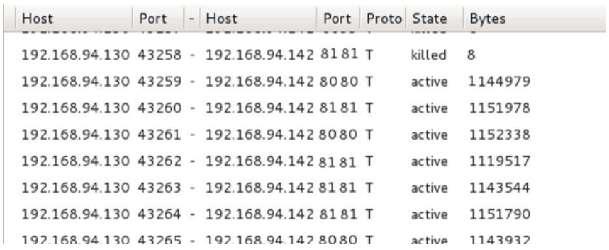
IV. RESULTS & DISCUSSION

A. Experiment Results

We have different Open source controllers such as NOX/POX - Floodlight - ODL - Ryu - ONOS and others cited in the article [15]. Table II illustrate the comparison between three open source controllers : ODL - ONOS and Ryu. We can highlights that ODL has a very good features, must be followed by ONOS then Ryu.

In this paper experiment we interpret only the first scenario, which was to capture traffic and information for identification of southbound interface connection. Figure 3 shows the relative traffic to the ODL interface , it is thanks to the ARP spoofing all traffic destined to the controller went through Kali-Linux.

In order to identify the practical problems related to our MITM attack in a League network, we make comparisons of the structure in a simulated environment using tools like Ettercap, SSLstrip showing it in table III, in the same sense as we have also run the attack SQL injection to steal the passwords of the session controller.



Host	Port	-	Host	Port	Proto	State	Bytes
192.168.94.130	43258	-	192.168.94.142	8181	T	killed	8
192.168.94.130	43259	-	192.168.94.142	8080	T	active	1144979
192.168.94.130	43260	-	192.168.94.142	8181	T	active	1151978
192.168.94.130	43261	-	192.168.94.142	8080	T	active	1152338
192.168.94.130	43262	-	192.168.94.142	8181	T	active	1119517
192.168.94.130	43263	-	192.168.94.142	8181	T	active	1143544
192.168.94.130	43264	-	192.168.94.142	8181	T	active	1151790
192.168.94.130	43265	-	192.168.94.142	8080	T	active	1143932

Fig. 3. Connection between the controller and interface web ODL

Figure 3 shows that the "192.168.94.130" address corresponds to the Mininet emulator that is launching connection with the ODL interface and "192.168.94.142" address represents the ODL controller. This figure 3 shows that the ODL interface does not use port 80 for HTTP traffic but uses ports 8080 and 8181. These ports use the HTTP traffic to establish a connection with the controller and not HTTPS. This means we can sniff the identification information connection of the ODL interface.

Capture 4 shows two communication are perform using the protocol ARP spoofing, in the first one there is an exchange of communication between the interface ODL and KALI-linux, in the second one the communication is between kali-linux and the MININET emulator, using the port 8181. At this stage we can say that the malicious machine communicate with both parts as it was explained in the scenario 1.

All these captures are intended to show that the connection between the web interface of ODL controller and the emulator is well-established, since it must retrieve the identification information of interface connection ODL in a machine that is malicious, for that the machine collect information on both machines ODL, MININET. We show also that the connection between the control layer and the infrastructure layer communicates with OpenFlow Protocol. Finally, this experience has shown that this small vulnerability can lead to an attack, where the attacker can exploit it with bad attention, so we have to solve the vulnerabilities.

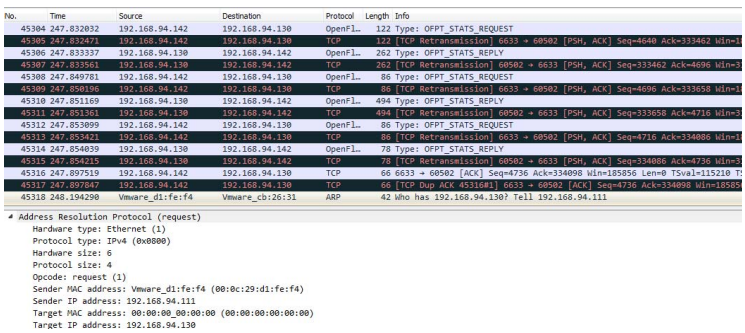


Fig. 4. Capture Wireshark showing Communication Information

TABLE II
COMPARISON OF OPEN SOURCE CONTROLLERS

	ONOS	**Ryu**	**OpenDayLight**
Langage	JAVA	Python	JAVA
Documentation	Good	Fair	Very Good
GUI	Web Interface	Intial phase	Web Interface
Destributed/centralized	Destributed	Centralized	Destributed
SouthBound APIs	OpenFlow 1.* , NetConf	OpenFlow 1.* , NetConf, OfConfig	OpenFlow 1.* , NetConf, YANG, OVSBD, PCEP, BGP/LS, LiSP, SNMP
NourtBound APIs	REST API	REST API (SB APIs)	REST API
Virtualization	Mininet & OpenVswitch	Mininet & OpenVswitch	Mininet & OpenVswitch
Automation	Yes	Yes	Yes
TLS Support	Yes	Yes	Yes
Network Monitoring	Yes	Yes	Yes
Open Source	Yes	Yes	Yes
Authantification	Yes	NO	Yes
Vulnerable	Yes	Yes	Yes

TABLE III
COMPARING TWO METHODS OF MITM ATTACKS

	Passive MITM
SSLstrip	* Attacker can monitor the connection and observes the transmitted information * Attacker can extract traffic patterns or any useful information such as location or participants identities by monitoring traffic * Attacker can see the identification session clearly with SSLstrip methods
	Active MITM
Ettercap	* Attacker pretends to be authenticity * Attacker captures and retransmits the message * Attacker alters deletes or recorded portions of the original message * Attacker makes the resource unavailable to the legitimate user
Results	***SSLstrip + Ettercap = Difficult MITM***

B. Discussion and Defense Measure

In order to prevent hackers to infiltrate in the internal networks,the figure 5 present defensive measure using secure technique, probably be usefully to integrate the application like Bring Your Own Device (BYOD) [17], introduce very effective security policies and establish awareness sessions for improved security for users. We need to define solutions to reducing the impact of the popular vulnerabilities, for that it necessary to secure the SDN controller. Software Defined Security Network (SDSN) includes a series of measures as the policy, detection and security application [3], [6], with a complete suite of products that centralize and automatically secure SDN.

It is very difficult to detect the MITM attack. There are some tools to treat, detect and prevent attacks such as ARP spoofing [10], [16] has to infiltrate the network. Snort, ArpAlert and ArpwatchNG [2], [7], [8] are very known solutions to solve this type of problem. ArpwatchNG is similar to ArpAlert, it has been watching MAC addresses used on the network. But the best solution is snort ,which is based on an IPS intrusion prevention system and an ARP spoof preprocessor [21]. On the other hand, we are surprised that the ODL Web interface is updated on HTTP and not HTTPS. ODL developers need to fix this vulnerability and apply an HTTPS approach for all network traffic.

V. CONCLUSION & PERSPECTIVES

First of all, controller vulnerabilities are a significant issue that SDN designers must address in the upcoming years. Evidence shows that we can suggest an absolute security for SDN controller.

In this work, an attack was correctly carried out on controller SDN; the attacker can control the inspector and compromise the whole network. Our work focused to study a vulnerabilities on the platform ODL, caused by an attacker who carries out an attack MITM on controller SDN.

We have presented a comparison of opensource controllers ODL - ONOS - Ryu, then described two methods of MITM attacks and presenting potential security issue in SDN scurity threats and attack surfaces.

Seem likely to confirm our scenarios hypothesis that the communication between the control layer (ODL) and infrastructure layer (mininet) is not secure from the southbound interface.

In addition, we have also proposed an effective countermeasure to prevent attacks on the League Monitor's monitoring service.

We hope that our research will serve as a base for future studies on utilizing more methods to deal with further investigation on vulnerabilities that are particular to the SDN

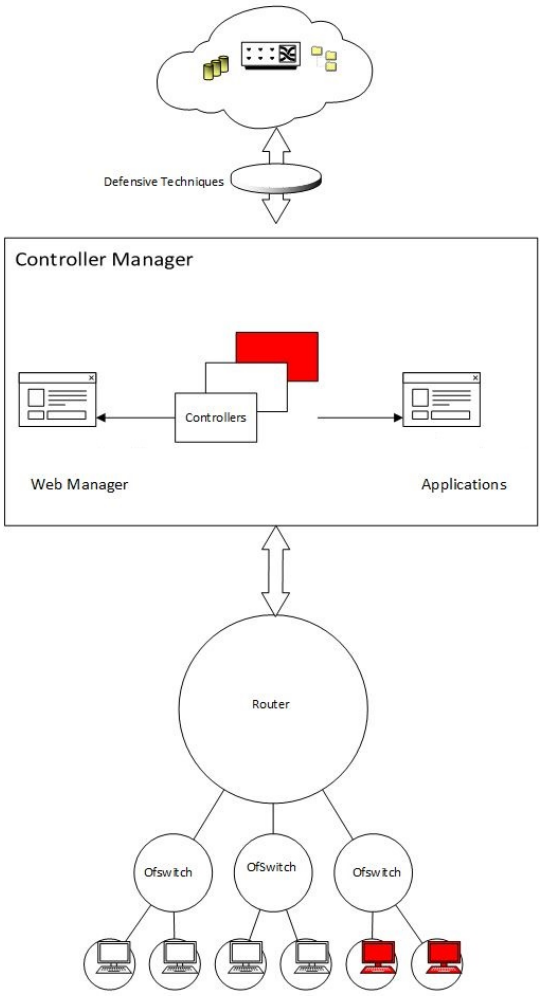


Fig. 5. Defensive measure

controllers and architectures. Finally, we will propose an effective countermeasure program to prevent & detect attacks.

REFERENCES

[1] Sdx Central. SD-WAN . https://www.sdxcentral.com/sd-wan/definitions/software-defined-sdn-wan/, 2017. [Online; accessed june-2017].

[2] Cisco. SNORT. https://www.snort.org/, 2017. [Online; accessed june-2017].

[3] Marc C Dacier, Sven Dietrich, Frank Kargl, Hartmut König, et al. Network attack detection and defense–security challenges and opportunities of software-defined networking. *Dagstuhl Reports, vol*, 6:1–28, 2017.

[4] Linux Fondation. opendaylight. https://www.opendaylight.org/#welcome, 2017. [Online; accessed june-2017].

[5] Ryu SDN Framework. Ryu Controller. https://osrg.github.io/ryu/, 2017. [Online; accessed August-2017].

[6] Diego Kreutz, Fernando Ramos, and Paulo Verissimo. Towards secure and dependable software-defined networks. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 55–60. ACM, 2013.

[7] Linux. Arpalert. http://www.arpalert.org/arpalert.html, 2017. [Online; accessed june-2017].

[8] Linux. Arpwatch. https://fr.wikipedia.org/wiki/Arpwatch, 2017. [Online; accessed june-2017].

[9] Kali Linux. Kali Linux. https://www.kali.org/, 2017. [Online; accessed june-2017].

[10] Mohammad Z Masoud, Yousf Jaradat, and Ismael Jannoud. On preventing arp poisoning attack utilizing software defined network (sdn) paradigm. In *Applied Electrical Engineering and Computing Technologies (AEECT), 2015 IEEE Jordan Conference on*, pages 1–5. IEEE, 2015.

[11] PISK MITM. TLS MitM attack on OpenFlow switch using Ryu controller app . https://github.com/m-kostrzewa/PSIK-MitM, 2017. [Online; accessed june-2017].

[12] Tri-Hai Nguyen and Myungsik Yoo. Attacks on host tracker in sdn controller: Investigation and prevention. In *Information and Communication Technology Convergence (ICTC), 2016 International Conference on*, pages 610–612. IEEE, 2016.

[13] ONF. ONOS Controller. http://onosproject.org/, 2017. [Online; accessed August-2017].

[14] Lisa Schehlmann, Sebastian Abt, and Harald Baier. Blessing or curse revisiting security aspects of software-defined networking. In *Network and Service Management (CNSM), 2014 10th International Conference on*, pages 382–387. IEEE, 2014.

[15] SdxCentral. SDN Archetecture. https://www.sdxcentral.com/sdn/definitions/inside-sdn-architecture/, 2017. [Online; accessed Februar-2017].

[16] Sandeep Singh, RA Khan, and Alka Agrawal. Prevention mechanism for infrastructure based denial-of-service attack over software defined network. In *Computing, Communication & Automation (ICCCA), 2015 International Conference on*, pages 348–353. IEEE, 2015.

[17] BYOD Team. BYOD. https://fr.wikipedia.org/wiki/Bring_your_own_device, 2017. [Online; accessed june-2017].

[18] Mininet Team. MININET. http://mininet.org/, 2017. [Online; accessed june-2017].

[19] Hao Tu, Weiming Li, Dong Li, and Junqing Yu. A scalable flow rule translation implementation for software defined security. In *Network Operations and Management Symposium (APNOMS), 2014 16th Asia-Pacific*, pages 1–5. IEEE, 2014.

[20] Network Functions Virtualisation. Introductory white paper. In *SDN and OpenFlow World Congress, Darmstadt, Germany*, 2012.

[21] Tianyi Xing, Zhengyang Xiong, Dijiang Huang, and Deep Medhi. Sdnips: Enabling software-defined networking based intrusion prevention system in clouds. In *Network and Service Management (CNSM), 2014 10th International Conference on*, pages 308–311. IEEE, 2014.

[22] Qiao Yan, Qingxiang Gong, and Fang-an Deng. Detection of ddos attacks against wireless sdn controllers based on the fuzzy synthetic evaluation decision-making model. *Ad Hoc & Sensor Wireless Networks*, 33(1-4):275–299, 2016.