http://dx.doi.org/10.14236/ewic/icscsr19.5

BLOSTER: Blockchain-based System for Detection of Fraudulent Rules in Software-Defined Networks

Abdelouahid Derhab
King Saud University
Saudi Arabia
abderhab@ksu.edu.sa

Mohamed Guerroumi
USTHB University
Algeria
guerroumi@gmail.com

Leandros Maglaras
De Montfort University
UK
leandros.maglaras@dmu.ac.uk

Mohamed Amine Ferrag
Guelma University
Algeria
ferrag.mohamedamine@univ-guelma.dz

Mithun Mukherjee
Guangdong University
of Petrochemical Technology
China
m.mukherjee@ieee.org

Farrukh Aslam Khan
King Saud University
Saudi Arabia
fakhan@ksu.edu.sa

This work proposes BLOSTER, a lightweight and accurate detection system, which leverages the blockchain technology to detect any tampering attempt with the OpenFlow rules of the software-defined networks. We test BLOSTER on an experimental platform combining blockchain and software-defined networking technologies. Security analysis and performance results demonstrate that BLOSTER ensures full detection of fraudulent rules within short detection time.

Blockchain, Software-Defined Network, OpenFlow

1. INTRODUCTION

Software-defined networking (SDN) (Farhady et al. (2015)) decouples the networking hardware from its control mechanism, by moving the control of lookup tables stored in the network devices to a central location, which allows easy control and management. The SDN includes two main components: (1) *SDN controller*, it uses protocols such as OpenFlow (openflow (2019)) to send flow rules to the switches. Basically, SDN controller tells the switches where to send data packets, and (2) *Virtual Switch (vSwitch)*, it is an application that interconnects multiple virtual machines of the same or different hypervisors, moreover, it interconnects these virtual machines with other physical switches.

Motivations and our contributions: In this work, we propose a security architecture that deals with the issues that arise due to the adoption of the software-defined technology, e.g., an adversary that injects fraudulent flow rules, which prevent correct routing of information. Specifically, we propose BLOSTER, which can detect in a short time any tampering with the OpenFlow rules.

Earlier works like Veriflow (Khurshid et al. (2013)), FortNOX (Porras et al. (2012)), and FlowChecker (Al-Shaer et al. (2010)) detect this attack by employing a heavyweight analysis process such as: analyzing the flow table to detect rule conflicts, and analyzing all switch configurations using model checking and binary decision. Differently from the above mentioned works, BLOSTER is lightweight in the sense that it only compares the traffic flow rule that is originated from the vSwitch with the one sent by the SDN controller.

Attack model: As presented in Fig. 1(a), we consider an attacker A can launch the following attacks:

- *Unauthorized Access to vSwitch*: Similar to the physical switch, Vswitches could be misconfigured in a way that allows a device impersonating another device. An attacker A can masquerade as the SDN controller and insert fraudulent rules in the flow tables of other vSwitches.
- *Unauthorized Access to SDN controller*: When an attacker A impersonates an SDN controller, it can gain access to the target's sensitive

information, including manipulating the SDN by using false identities.

- *Man-in-the-middle attack between switch and controller*: By spoofing the identities of the two nodes, an attacker A can send fake flow rules to the vSwitches.

2. BLOSTER ARCHITECTURE AND ITS IMPLEMENTATION

BLOSTER is based on the blockchain technology. The flow rules, which are generated from the controller, are stored in a verifiable and immutable database. The blockchain is a sequence of blocks that are produced and signed by the members of the network (Ferrag et al. (2019)). All new blocks are broadcasted to other peers and upon validation are appended to the blockchain. Once recorded, the data in any given block cannot be changed without alteration of all subsequent blocks. Also, the data exists in multiple hosts at the same time, so any changes would be rejected by the peer's hosts. As shown in Fig. 1(b), BLOSTER is carried out in the following way:

- Upon receiving a request from an application, the SDN controller sends the corresponding flow rules to the vSwitches. The SDN controller is a member of the blockchain having extra privileges, being the only node that can create new blocks. It hashes the flow rules and puts them in a block that is distributed to the other nodes of the blockchain.
- When the flow rules reach the vSwitch node, the latter updates its flow table and saves the rules in a log file.
- The Firewall collects the vSwitch log and accesses the blockchain to obtain the flow rules sent by the controller.
- If the firewall finds that the two set of rules, from vSwitch and blockchain, are not similar, it notifies the Administrator to take the appropriate countermeasures.

BLOSTER is implemented, as shown in Fig. 1(c), using the following components: (a) **Private cloud**, implemented using Openstack (openstack (2019)), (b) **Blockchain**: It uses Multichain (multichain (2019)), derived from Bitcoin Core (bitcoin-core (2019)), to implemente a private blockchain. JSON (json (2019)) is used to create blocks. The role of the blockchain is to save all the operations transmitted from the SDN controller. Multichain ensures that the activity of the blockchain is only visible to the chosen participants, and it provides read and write privileges on the transactions, (c) **SDN controller**: ONOS (onos (2019)) is used to implement the SDN controller. It provides the control plane of the network by managing its components such as routers, switches, and links, (d) **Mininet**: is a network emulator, which creates a network of controllers, switches, virtual hosts, and links. It allows creating an SDN prototype to simulate a network using switches supporting OpenFlow.

As shown in Figure 1(c), the Insert() program captures the traffic sent from the ONOS controller to the vSwiches, which allows getting the flow rules of the ONOS controller, and storing them on the blockchain.

3. SECURITY ANALYSIS

The security of BLOSTER is discussed with respect to the following attacks:

- *Unauthorized Access to SDN controller*: the SDN controller is assumed to be located in a private cloud, and is only reachable from a single host, which applies an authentication and access control mechanism as in mattos et al. (2016). Therefore, the SDN controller cannot generate fraudulent flow rules.
- *Man-in-the-middle attack between switch and controller*: As the SDN controller does not generate fraudulent flow rules, and it is the only one that has the right to create blocks in the blockchain, the created legitimate flow rules are legitimate. If the flow table of the vSwitch is poisoned with fraudulent rules, the firewall will eventually detect this attack after comparing the rules stored in the blockchain and the vSwitch logs.
- *Unauthorized Access to vSwitch:* If forged flow rules are injected in the flow table of the vSwitch, this attack can be detected by the firewall.
- *Blockchain poisoning:* Under this attack scenario, we assume that an attacker impersonates as an SDN controller and injects the same flow rule in both the vSwitch and the blockchain. This scenario cannot happen, as the blockchain is only updated by the SDN controller, which has a unique private key.

4. PERFORMANCE EVALUATION AND INSIGHTS

We evaluate the performance of BLOSTER by testing the effect of injecting false rules into the network. To do so, we disconnect the SDN controller and inject the rules at the switch-level. From Table 1, the important findings are as follows:

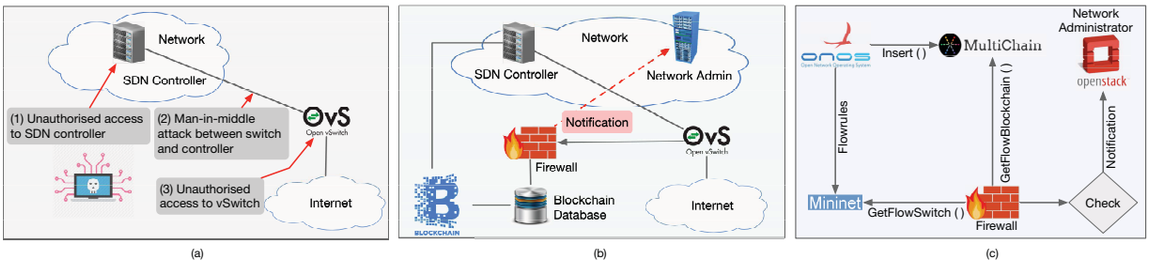

Figure 1: *(a) Threat model considered in the network. (b) BLOSTER architecture and (c) its implementation components.*

Table 1: *False Rule Detection Performance of BLOSTER*

Number of injected rules	Detection time (sec)	Detection rate (%)
10	2.40	100%
50	2.43	100%
100	2.44	100%
200	2.45	100%
500	2.54	100%
750	2.57	100%
1000	2.64	100%
1500	2.70	100%
2000	2.83	100%

- BLOSTER achieves a detection rate of 100% of fraudulent flow rules.
- The detection time of BLOSTER is very low, and scalable with respect to the number of injected rules.

As a part of future work, we plan to adapt BLOSTER to the multi-SDN controller environment. Moreover, it would be interesting to leverage the blockchain technology to prevent injection of fraudulent flow rules in the flow tables, instead of only detecting them.

REFERENCES

H. Farhady, H. Lee, and A. Nakao, Software-defined networking: Asurvey,Computer Networks, vol. 81, pp. 7995, 2015.

`https://www.sdxcentral.com/sdn/definitions/what-is-openflow/`

A. Khurshid, X. Zou, W. Zhou, M. Caesar, and P. B. Godfrey, Veriflow:Verifying network-wide invariants in real time, the 10th USENIX Symposium on Networked Systems Design andImplementation (NSDI), 2013, pp. 15-27.

P. Porras, S. Shin, V. Yegneswaran, M. Fong, M. Tyson, and G. Gu, "A security enforcement kernel for OpenFlow networks," Proceedings of the first workshop on Hot topics in software defined networks. ACM, 2012, pp. 121-126.

E. Al-Shaer, and S. Al-Haj, "FlowChecker: Configuration analysis and verification of federated OpenFlow infrastructures," Proceedings of the 3rd ACM workshop on Assurable and usable security configuration. ACM, 2010, pp. 37-44.

M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, and H. Janicke, Blockchain technologies for the internet of things: Researchissues and challenges,IEEE Internet of Things Journal, vol. 6, no. 2,pp. 2188 - 2204, 2019.

`https://www.openstack.org/`

`https://www.multichain.com/`

`https://bitcoin.org/en/bitcoin-core/`

`https://www.jsonrpc.org/specification`

`https://www.opennetworking.org/onos/`

D. M. F. Mattos and O. C. M. B. Duarte, Authflow: authentication andaccess control mechanism for software defined networking, Annals of Telecommunications, vol. 71, no. 11-12, pp. 607 - 615, 2016.