Proceedings of 8th IOE Graduate Conference
Peer Reviewed
ISSN: 2350-8914 (Online), 2350-8906 (Print)
Year: 2020 Month: June Volume: 8

Security Analysis of SDN using Blockchain Technology

Prakash Pandey [a], Rupesh Kumar Sah [b]

[a, b] *Department of Electronics and Computer Engineering, Pashchimanchal Campus, IOE, TU, Nepal*
Corresponding Email: [a] erprakash.unik@gmail.com, [b] rupeshsah@ioe.edu.np

Abstract

Software-Defined Networking (SDN) is a network architecture that improvises a conventional way of networking in terms of control, scalability, security, and availability. Currently, limited researches are being conducted in the field of SDN security. Blockchain is one of the important techniques to overcome SDNs security issues. Cryptographic techniques are useful for SDN but these techniques has an additional burden of computing time. Blockchain technology is more sophisticated than cryptographic techniques due to their transparency, immutability, decentralize, and computational efficiency. In SDN, even though, the control plane is separated from the underlying forwarding plane, SDN is susceptible to many security challenges like Denial of Service (DoS) attack, Distributed DoS (DDoS) attack, and Brute force attack. In this paper, we have analyzed some security issues of SDN by using vulnerability assessment tools. Mininet emulator is used for simulating the SDN network. OpenDaylight controller is used as SDN controller. Blockchains are distributed ledgers that maintain records of packets transmitted through SDN data layer and avoids invasion of any false flow rules on it by using its monolithic secure mechanism.

Keywords

Software Define Network (SDN), Blockchain, Mininet, OpenDaylight, Security

1. Introduction

Software Defined Networking (SDN) is an emerging networking paradigm[1] that greatly simplifies network management tasks[2]. SDN aims to make networks agile, flexible and improve network control by enabling enterprises and service providers to respond quickly to changing business requirements. Network functions can be run on less-expensive off-the-shelf hardware, reducing capital expenditure. Enterprises can reduce operational expenditure on IT services by supporting automation and algorithm control through increased programmability of network elements to make it simple to design, deploy, manage, configure and scale networks. SDN approaches and technologies help organizations rapidly deploy new, fluid, and flexible applications, services and infrastructure to quickly meet their changing requirements.

The core of an SDN network is Controller. Controller is an application in software defined networking architecture that manages flow control for improved network management and application performance. SDN controllers direct traffic according to forwarding policies that a network operator puts in place, thereby minimizing manual configurations for individual network devices. If a Controller is compromised, then the whole network under the SDN Controller is made vulnerable. SDN Controller can be compromised in three ways:

- Due to malicious software errors or bugs in the controller software system.
- Due to threats arising from malicious or compromised applications driving the controller.
- Threats from the underlying network devices such as the OpenFlow switches.

The centralized nature of SDN makes it vulnerable to DoS attacks[3] which can disable the whole network or a component of the network and can degrade its performance. DDoS attacks are a way that attackers make certain online services unavailable by flooding them with excessive fake traffic. The traffic consumes bandwidth and resources, which causes the SDN controller shut-down. DDoS attacks are one of the most common attacks on SDN controller.

Blockchain is a chain of blocks and it has its own specification[4]. These blocks are transparent, immutable and decentralized by carrying the data, same Block hash as well as previous Block hash. Blockchain is a temper-proof, distributed data structure that is replicated and shared among the members of a network[5]. This data structure acts as a log whose elements (Blocks) are batched into time stamped entries, uniquely identified either on the Block's content or its header, contains a subset of the overall transactions record made by all interconnected nodes with proper access to the system and includes a reference to the preceding blocks hash. This method forms a link between blocks that connects to form a chain, the Blockchain[6].

This paper mainly focuses on the security issues of SDN where the issues on controller may lead the entire system to go down. The controller is highly vulnerable to DoS, DDoS and Brute-Force attack, as the software of Controller runs on open source operating system (OS). Hence, the chance of attack in SDN is high and there can be a risk on confidentiality, integrity and availability of SDN network. Blockchain-based monolithic secure mechanism effectively addresses security issues of SDN[7]. Software Defined Networking is regarded as an emerging paradigm that provides better visibility and controllability with security to the network for better performance and efficiency.

2. Software Defined Network

SDN has arisen from these service-focused necessities[8]. The control layer is moved out of the individual network nodes and into the different, centralized controller. Network Operating System (NOS) is controlling SDN switches by collecting data using the API and maneuver their forwarding plane, providing an abstract model of the network topology to the SDN controller hosting the applications. The controller can utilize complete knowledge of the network to improve flow management and support service-user requirements of scalability and flexibility.

Briefly, in traditional networking, the control planes(CP) and the data planes(DP) are collocated on devices to ensure decentralized network control[9]. While in SDNs, the DPs and CPs are separated with a centralized controller controlling multiple DPs while supporting a southbound application programming interfaces (API) to the DPs and a northbound API to the SDN applications.

3. Security Issues in SDN

It is important to understand that SDN is also susceptible to security attacks. Figure 1 indicates the SDN security vectors at the three layers, Data plane layer; Controller layer and SDN Application layer[10]. One of the basic security issues that needs attention is on the controllers. If one does not manage the control panel right, the centralized controller may be a potential single point of attack and failure. The controller is the core component in SDN[11]. It defines the data flow that takes place in the data plane. It is the "brain" of the network. Hence, when the controller is attacked and compromised, it will definitely affect the network badly.

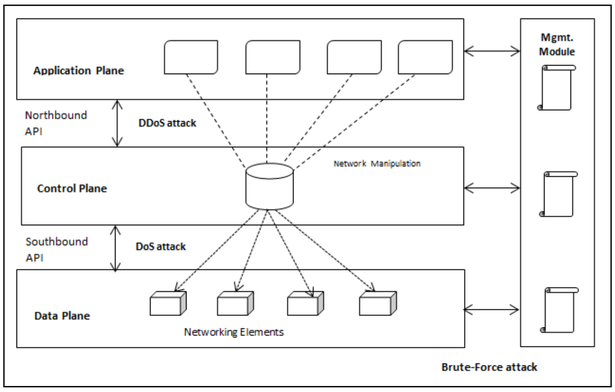


Figure 1: SDN architecture with its attack vector

4. Related Work

Prior to the selection of study area, different literatures related to the study of Software Define Networking and Distributed ledger technology were reviewed and analyzed. Currently, only limited researches are being conducted on SDN security issues such as vulnerability, risk, threats and attacks. With the extensive adoption of Blockchain, some security issues of SDN are exposed and imperatively studied.

At the time of reviewing and analyzing different literatures, we found that the research gap exists on security issues and decided to performed the following tasks:

- We have conducted DoS/DDoS and Brute force attack simulation on SDN network by using hping3 and xhydra respectively to analyze the

impact on SDN and its performance before and after integrating with Blockchain.
- Finally, we have developed Blockchain-based monolithic secure mechanism to enhance the security of SDN.

None of these above mentioned tasks were performed previously by any researchers, our approach is to enhance the SDN security

5. Methodology

SDN controller was developed under the OpenDaylight framework to manage SDN network. OpenFlow switches in the data network establish their TCP connections with OpenDaylight on the controller node. Then, Open vSwitches (OVS) are connected to OpenDaylight controller to manage OVS. Figure 2 illustrate the proposed research methodology.

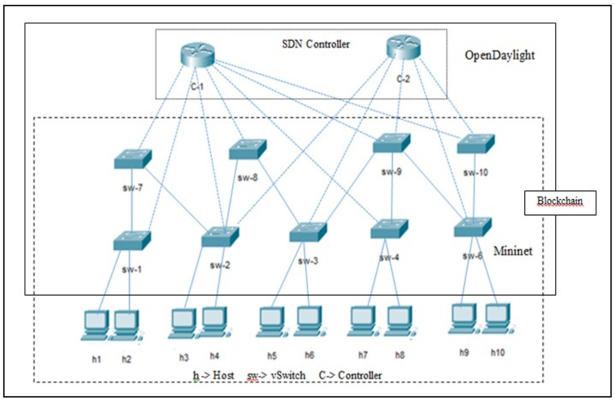
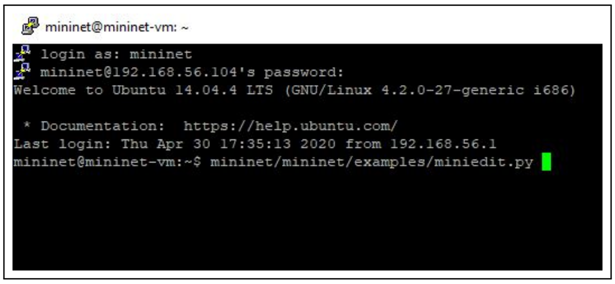


Figure 2: Research Methodology

All OpenFlow vSwitches are developed under the Mininet by employing different network topologies and traffic load. Mininet is most widely used netwok emulation tool in SDN network. Opendaylight SDN controllers were used in this experiment as shown in figure 2. Controllers C-1 and C-2 are running over VM 6.1.6 and connected through virtual links. Controllers over each VM is connected to a Mininet instance, emulating an area network that contains network switches and host devices as Mininet hosts. We have leveraged connectivity between controller instances to share Blockchain information for control plane synchronization using off band channel implementation through virtual links between the VMs. Each Opendaylight controller instance use hyper fabric for the implementation of Hyper-ledger and hping3 tool to generate the traffic of DoS and DDoS flooding attack. Using the number of flows different attacks such as TCP/SYN flood, UDP flood, ICMP flood are launched and the system is simulated.

5.1 Experimental Settings

We tested our experiments on machine with an Intel Core i7-2600 / 3.40 GHz / 8 cores processor and 8 GB of RAM with Linux Operating System. MiniEdit running over Mininet 2.2.2 is used to emulate the network operation as shown in figure 3. Mininet is easy for testing and exploring software-defined networks, allowing us to create a network of virtual hosts, switches, controllers, and channels. Mininet components such as hosts, switches, and controllers are truly virtual. This emulator comes with open source code with which you can create a realistic virtual network on real hardware. Any code developed in Mininet can also be run on a real network without any fixes.



```
 login as: mininet
 mininet@192.168.56.104's password:
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic i686)

 * Documentation:  https://help.ubuntu.com/
Last login: Thu Apr 30 17:35:13 2020 from 192.168.56.1
mininet@mininet-vm:~$ mininet/mininet/examples/miniedit.py
```

Figure 3: MiniEdit run from Mininet

We set up the experimental SDN topology model in such a way that it is possible to simulate DoS, DDoS and Brute force attacks. After creating a SDN topology on MiniEdit as shown in figure 3. In MiniEdit we set up all configurations and then start run. First we ping all host by executing 'pingall' command. All hosts are reachable, and then we executed 'xterm h2' command to open the terminal of node h2. The 'xterm' command is used to open individual terminals for hosts. From h2 terminal, DDoS attacks started by executing 'hping3' command to the random source as shown in figure 6. Hping3 generates an abnormal traffic in SDN network. DDoS causes reduction or complete disruption of SDN services. Sometime, after we tested connectivity h6 to h3 and h10 to h5 by executing 'h6 ping h3' and 'h10 ping h5' command on MiniEdit terminal. We continuously monitored connectivity on screen but connectivity was disrupted and destination host unreachable message appeared on screen. After performing we again restarted the network and executed 'pingall' command. Then, we got network running successfully. At the same time we also monitored resource consumption of host and node. Again we executed '/test.sh' to send normal traffic from node h6. Test.sh is just a script of random packet of normal traffic for random duration. SDN network executed './test.sh' command in order to pass those random packets of normal traffic normally.

Similarly, we simulated for Brute force attack too. This attack happens on non SDN elements. With password guessing or brute force, an unauthorized user gains the access of network. At this time, we run xhydra on h2 while h7 is the target for the attack. Among several Brute-Force attacking tool, we choose xhydra. This help us unauthorized access to h7 in our experiment. By the help of xhydra we get the plane text password of h7. When we get the plain text password, we need to confirm whether that password is genuine or not. To confirm this we used plane text password to access h7. We got access by using plane text password. Brute-Force attack is quite difficult as compare to DoS and DDoS attack. All the above experiment was performed on SDN network before integration with Blockchain.

We install Hyperledger fabric on another machine VM with same specification as previous one. Hyperledger fabric is the modular Blockchain framework that has become the de facto standard of enterprises Blockchain. After configuring the SDN topology on Blockchain platform, we repeat the same experimental process as done in SDN network earlier to justify the research objectives. We found that the network was not compromised during attack.

6. Result and Discussion

6.1 SDN Topology

In this work, Mininet emulator was used for simulating SDN topology. The OpenDaylight OpenFlow controller was used to manage the network of SDN. All the OpenFlow switches in figure 3 established TCP connections with OpenDaylight on the controller node.

We model the experimental SDN in such a way that it makes it possible to simulate DoS, DDoS and Brute force attacks. Brute force attack happen on non-SDN element to gain access to the SDN. It is noteworthy that the network emulation used a simplified SDN, which allows carrying out all the necessary experiments. In fact, it demonstrates the ability to simulate attacks on a network of this size.

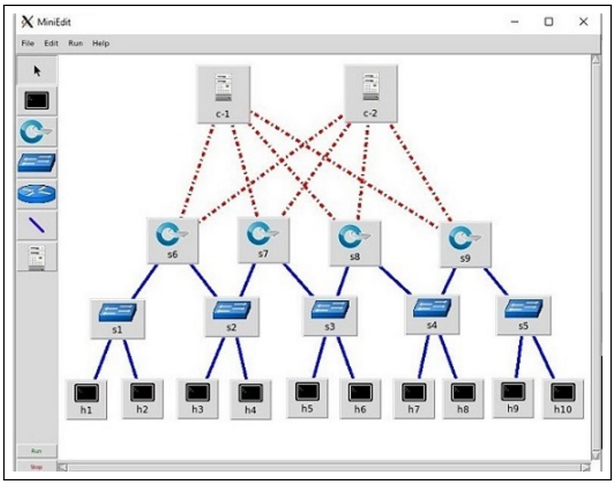


Figure 4: Software Define Network Topology

Although the network may not be large with a lot of components, the emulated topology consists of ten host and nine switches and two controllers as shown in figure 4. Each switches are connected to a C-1 and C-2 controllers. Centralized controllers maintain the forwarding rules in the respective flow tables. So that the controllers can make forwarding decisions based on forwarding rules defined in the respective flow tables of Open OVS.

6.2 DoS and DDoS attack on SDN

On hosts we designed a system to attack a stable SDN network, For this we start sending random 100 packet per second by executing sudo hping3 –faster –rand-source 10.0.0.1 command, which floods the attacked host with many packets with different source IP addresses. As a result, it can be seen that each of the attacking hosts is able to create many malicious packets in a fraction of seconds.

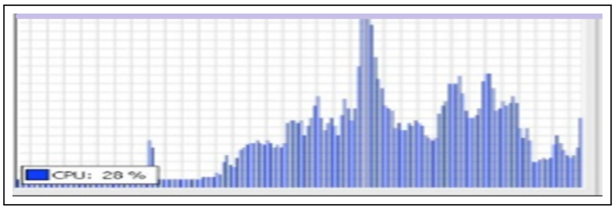
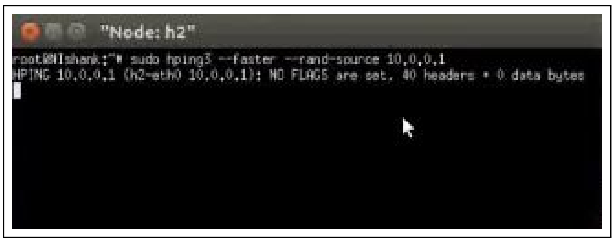


Figure 5: Network traffic during a DDoS attack

The network was tested with normal traffic, DoS and DDoS attack. Network showed very minimum traffic when no external traffic was introduced to the network accounting for the communication protocol between the nodes. Normal Random traffic packet test.sh was flooded from Node h6 which utilized the resources of the network and the traffic was increased to about 25 percentage. When 'hping3' was executed network traffic raise highly and more resource were utilized as seen in figure 5.

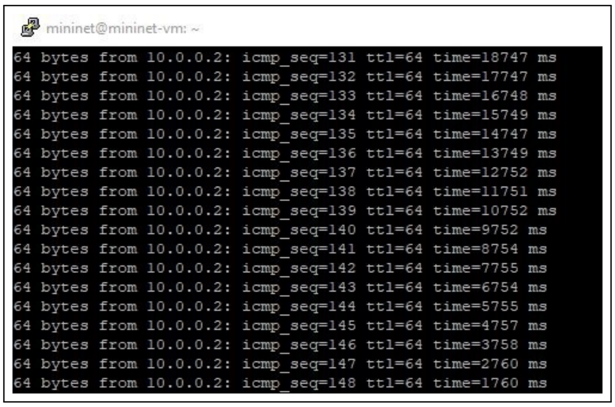
The 'xterm' command is used to open individual terminals for hosts. From h2 terminal, DoS and DDoS attacks was started by executing 'hping3' command to the random source h1 with IP address 10.0.0.1. 'hping3' generates a abnormal traffic in SDN network as seen in figure 6.



```
root@Nishank:~# sudo hping3 --faster --rand-source 10.0.0.1
HPING 10.0.0.1 (h2-eth0 10.0.0.1): NO FLAGS are set, 40 headers + 0 data bytes
```

Figure 6: DDoS attack using hping3

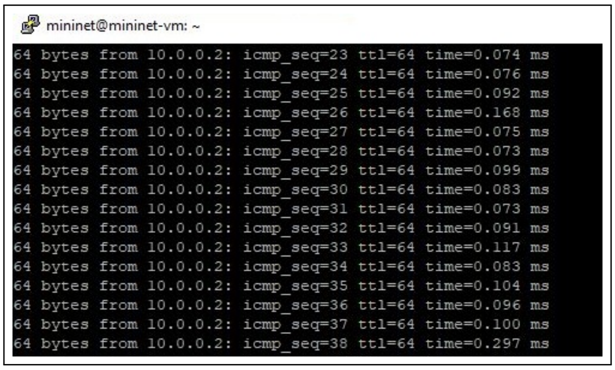
DDoS causes reduction or complete disruption of SDN services as seen in figure 7. From figure 7, we can justify that when abnormal random packets were flooding on SDN network we observed minimum 1760 ms to maximum 18747 ms latency and SDN performance was reduced.



```
64 bytes from 10.0.0.2: icmp_seq=131 ttl=64 time=18747 ms
64 bytes from 10.0.0.2: icmp_seq=132 ttl=64 time=17747 ms
64 bytes from 10.0.0.2: icmp_seq=133 ttl=64 time=16748 ms
64 bytes from 10.0.0.2: icmp_seq=134 ttl=64 time=15749 ms
64 bytes from 10.0.0.2: icmp_seq=135 ttl=64 time=14747 ms
64 bytes from 10.0.0.2: icmp_seq=136 ttl=64 time=13749 ms
64 bytes from 10.0.0.2: icmp_seq=137 ttl=64 time=12752 ms
64 bytes from 10.0.0.2: icmp_seq=138 ttl=64 time=11751 ms
64 bytes from 10.0.0.2: icmp_seq=139 ttl=64 time=10752 ms
64 bytes from 10.0.0.2: icmp_seq=140 ttl=64 time=9752 ms
64 bytes from 10.0.0.2: icmp_seq=141 ttl=64 time=8754 ms
64 bytes from 10.0.0.2: icmp_seq=142 ttl=64 time=7755 ms
64 bytes from 10.0.0.2: icmp_seq=143 ttl=64 time=6754 ms
64 bytes from 10.0.0.2: icmp_seq=144 ttl=64 time=5755 ms
64 bytes from 10.0.0.2: icmp_seq=145 ttl=64 time=4757 ms
64 bytes from 10.0.0.2: icmp_seq=146 ttl=64 time=3758 ms
64 bytes from 10.0.0.2: icmp_seq=147 ttl=64 time=2760 ms
64 bytes from 10.0.0.2: icmp_seq=148 ttl=64 time=1760 ms
```

Figure 7: Network traffic during DDoS attack

Consensus Protocol maintains ordering and validation among all network nodes in Blockchain by recording behaviours of SDN nodes(controllers and switches) that facilitates for easy auditing and debugging. RAFT is a consensus algorithm for managing replicated logs. For this verification, we simulate DDoS attack from host(h2) as shown in figure 6 and test connectivity among hosts by ping host h2 from host h1. We observed minimum 0.073 ms to maximum 0.297 ms latency as shown in figure 8. During DoS and DDoS attack there was effect on connectivity and packets transfer.



```
64 bytes from 10.0.0.2: icmp_seq=23 ttl=64 time=0.074 ms
64 bytes from 10.0.0.2: icmp_seq=24 ttl=64 time=0.076 ms
64 bytes from 10.0.0.2: icmp_seq=25 ttl=64 time=0.092 ms
64 bytes from 10.0.0.2: icmp_seq=26 ttl=64 time=0.168 ms
64 bytes from 10.0.0.2: icmp_seq=27 ttl=64 time=0.075 ms
64 bytes from 10.0.0.2: icmp_seq=28 ttl=64 time=0.073 ms
64 bytes from 10.0.0.2: icmp_seq=29 ttl=64 time=0.099 ms
64 bytes from 10.0.0.2: icmp_seq=30 ttl=64 time=0.083 ms
64 bytes from 10.0.0.2: icmp_seq=31 ttl=64 time=0.073 ms
64 bytes from 10.0.0.2: icmp_seq=32 ttl=64 time=0.091 ms
64 bytes from 10.0.0.2: icmp_seq=33 ttl=64 time=0.117 ms
64 bytes from 10.0.0.2: icmp_seq=34 ttl=64 time=0.083 ms
64 bytes from 10.0.0.2: icmp_seq=35 ttl=64 time=0.104 ms
64 bytes from 10.0.0.2: icmp_seq=36 ttl=64 time=0.096 ms
64 bytes from 10.0.0.2: icmp_seq=37 ttl=64 time=0.100 ms
64 bytes from 10.0.0.2: icmp_seq=38 ttl=64 time=0.297 ms
```

Figure 8: DDoS attack after Blockchain

Blockchain maintains chronological sequence of packets transmission informations. Each packets informations are kept in blocks, which can only recognized by the authorized hosts such that other nodes can notice and validate the change. The validated blocks are chained together. SHA-256 hashing function is used to index and retrieve transactions in database. Pre-image resistance hashing properties of Blockchain overcome Brute-force attack. IP address is added as transaction in block in Blockchain. Smart contracts are utilized for reporting white or black listed IP addresses across SDN nodes. SDN utilized flow rules to block DoS and DDoS attacks. whose flow needs to be stopped and whose flow need to be passed depending upon the flow policy of the network. When the controllers sends flow rule to SDN network element via OpenFlow protocol a copy of this flow rule is sent to the SDN node via Blockchain. Additionally, the Blockchain must ensure that no node should operating within its maximum capacity i.e. DoS and DDoS attacks. If any node compromise, rest of the other nodes automatically recognize the problem and simply stop executing the unacceptable activities by using network consensus algorithms.

From the experiment, we found that Blockchain overcome some security issues of SDN network efficiently as compared to the existing cryptographics techniques in term of time overhead, key propagation and revocation, modification and key storage and management. We also found that SDN network is emerging technology in application centric infrastructure(ACI) or convergence network but no more applicable in divergence network.

7. Conclusion

SDN has become an emerging technology; with the addition of the Blockchain technology leads to enhance the network performance. With its extensive adaptation, some security issue of SDN are exposed and imperatively studied which shows that SDN security issues such as vulnerability, risk ,threats and attacks can be reduced. Blockchain based SDN network nodes run a consensus protocol to achieve an agreement to generate a new block. Meanwhile, the data and transactions, in the new block are also confirmed due to its consensus protocol. Hence, the Blockchain-based monolithic secure mechanism avoids invasion of any false flow rules and effectively addresses security issues of SDN.

Acknowledgments

The authors are grateful to IOE, Pashchimanchal Campus, Lamachour Pokhara for providing this opportunity and lab arrangements.

References

[1] Danda B Rawat and Swetha R Reddy. Software defined networking architecture, security and energy efficiency: A survey. *IEEE Communications Surveys & Tutorials*, 19(1):325–346, 2016.

[2] Mohammad Mousa, Ayman M Bahaa-Eldin, and Mohamed Sobh. Software defined networking concepts and challenges. In *2016 11th International Conference on Computer Engineering & Systems (ICCES)*, pages 79–90. IEEE, 2016.

[3] BV Karan, DG Narayan, and PS Hiremath. Detection of ddos attacks in software defined networks. In *2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS)*, pages 265–270. IEEE, 2018.

[4] Dylan Yaga, Peter Mell, Nik Roby, and Karen Scarfone. Blockchain technology overview. *arXiv preprint arXiv:1906.11078*, 2019.

[5] Christos Tselios, Ilias Politis, and Stavros Kotsopoulos. Enhancing sdn security for iot-related deployments through blockchain. In *2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pages 303–308. IEEE, 2017.

[6] Rajat Kandoi and Markku Antikainen. Denial-of-service attacks in openflow sdn networks. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 1322–1326. IEEE, 2015.

[7] Weng Jiasi, Weng Jian, Liu Jia-Nan, and Zhang Yue. Secure software-defined networking based on blockchain. *arXiv preprint arXiv:1906.04342*, 2019.

[8] Sanjeev Singh and Rakesh Kumar Jha. A survey on software defined networking: Architecture for next generation network. *Journal of Network and Systems Management*, 25(2):321–374, 2017.

[9] Junaid Qadir, Nadeem Ahmed, and Nauman Ahad. Building programmable wireless networks: an architectural survey. *EURASIP Journal on Wireless Communications and Networking*, 2014(1):172, 2014.

[10] Deepak Singh Rana, Shiv Ashish Dhondiyal, and Sushil Kumar Chamoli. Software defined networking (sdn) challenges, issues and solution. *Int. J. Comput. Sci. Eng*, 7:1–7, 2019.

[11] Yosr Jarraya, Taous Madi, and Mourad Debbabi. A survey and a layered taxonomy of software-defined networking. *IEEE communications surveys & tutorials*, 16(4):1955–1980, 2014.