International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-12, October 2019

SIPAV-SDN: Source Internet Protocol Address Validation for Software Defined Network

Ramesh Chand Meena, Meenakshi Nawal, Mahesh Bundele

Abstract*: SDN technology is becoming every day more popular and big data centers and organizational networks have started deploying for its advantages. Current development of SDN network relies on target host IP address of packet and OFSwitches ignores checking of source host IP. SDN has separated control planes and data planes and OpenFlow protocol enabled switches are used as packet forwarding devices. The SDN controller controls flow of data packet through forwarding devices and when these are turned on, do not have any control and defense. The devices are not able to handle packet arriving from connected host. In this case, data packets of hosts are sent to the controller forwarding device for inspection and control packet creation for data packet and setting up required matching entries in flow table of forwarding device for such type of data packets generated by the hosts. The attackers can generate packets with Spoofed source IP address and perform various types of attacks. In this research paper, we offer a scheme as Source IP Address Validation for Software Defined Network (SIPAV-SDN) to check packet's source host IP address by binding source host IP Address and MAC address with switch port. It maintains a HostTable at Controller for verification of source host IP and MAC with switch port and only forwards the packets which have valid sources host IP address. We also simulated SIPAV-SDN with hybrid SDN network and experiment results have shown that it achieved 100% packet filtering accuracy for IP spoofed TCP, UDP and ICMP packet attacks. We used python programming language for RYU controller in Mininet network emulator.

Keywords: Source, IP, Address, Validation, SIPAV, SDN, Controller

I. INTRODUCTION

In traditional network forwarding devices have to handle incoming host packets based on destination IP address [15]. In SDN incoming packets are handled through flow entries in flow table of OpenFlow switch and flow entries are made using multiple fields of packet header. SDN networks also records traffic statistics. Programmability, elasticity and simplicity are provided by SDN environment to the network administrators and Software Defined Networks (SDN) usage is increasing in the various data centers and enterprise networks solutions to take the advantages of these opportunities, rectification is not possible.

The forwarding devices in OpenFlow network don't have any control & security polices initially and do not know how to handle data packet received from a host. Packets are forwarded by checking target host address but authenticity [8] of source host IP address is not checked and packets are forwarded to next hop till it reaches to its destination host. In this case, attackers may perform various IP spoofed attacks and block or fully occupy network resources and in this situation, resources will not be available for legitimate users. Without validity of sender's IP address, attackers will be able to conduct source IP spoofing attacks and desirable usage of network will not be achieved.

Access control lists (ACLs) are used to check source host address and filter packets for IP spoofing attacks in networks. ACLs are complex in nature and may create conflicts with existing security policies of the network. In [8] proposed a SAVSH for SDN environment with minimum deployment of OpenFlow protocol enabled switches. During study of SAVSH, we found that topology conversion into sink tree was limited to links between packet forwarding devices. Host details are not part of topology/sink tree and addition & deletion of host was not reflected in sink tree. Researchers used IP prefixes in flow rules and IP spoofing attacks are possible within the network with other host's IP in IP prefix filtering system.

In SDN-SAVI [32], researchers installed required flow entries in flow table of OpenFlow Switch to forward the AAM (Address Assignment Mechanism) packets to controller. Then controller updates the binding table and setups flow entries into flow table of the OpenFlow switch. The approach used IP address of a host and Switch port for binding and filtering packets. It has been noted that an attacker may generate forged AAM packets to mislead binding table generation. Approach used IP address of host and Switch Port ID for binding and verification of source IP address. Authenticity of source IP address does not provide full identification of a host without checking host's MAC and it may be less accurate. The binding between source host's IP address, MAC address and switch port ID provides finer filtering. Considering above, we propose a scheme for checking the packet's source host IP address as Source IP Address Validation for Software Defined Network (SIPAV-SDN). The scheme checks the authenticity of packet's source host IP address and only forwards the packets which have valid sources host IP address. SIPAV-SDN was simulated with hybrid SDN network also and system worked effectively as desired. The remaining sections of paper are organized as follows. Section 2 briefly discussed related works. Section 3 describes proposed SIPAV-SDN Controller.

Revised Manuscript Received on October 30, 2019.
***** Correspondence Author
Ramesh Chand Meena*, Research Scholar, School of Engineering & Technology, Poornima University, Jaipur, India. Email: rameshrmz@yahoo.com ORCID:0000-0002-2649-1384
Meenakshi Nawal, Associate Professor, Computer Science and Engineering, School of Engineering & Technology, Poornima University, Jaipur, India. Email: meenakshi.nawal@poornima.edu.in
Mahesh Bundele, Director & Principal, Poornima College of Engineering, Jaipur, India Email: maheshbundele@poornima.org

Section 4 describes implementation of SIPAV-SDN. Section 5 investigates the performance of SIPAV-SDN. Section 6 describes features and future works. Finally, Section 7 concludes the work.

II. RELATED WORKS

In [8] SAVSH to achieve maximum mitigation of spoofing attacks by adding inter domain module in SAVSH approach. The basic components of the SAVSH were Filtering Rule Generation, Checkpoint Selection & Topology Detection. In paper topology conversion into Sink Tree is limited to links between packet forwarding devices. Host details and changes in host were not reflected in sink tree. IP prefixes were used for setting up flow rules in approach and IP spoofing attacks are possible within network with other host's IP where a network is using IP prefix filtering mechanism. Authors, in [32] used IP address of host and Switch Port ID for binding and verification of source IP address. It is less accurate without host MAC binding and is not providing full host identification. The binding between source host IP address, MAC and switch port ID provides finer filtering. We addressed these issues by using the first ARP packet of host to update HostTable and setup flow entries related to the host. We used Host's IP, MAC, Switch PortID for setting up flow entries into flow table of OFSwitch.

III. PROPOSED SOURCE IP-ADDRESS VALIDATION FOR SDN (SIPAV-SDN)

3.1 Objectives of SIPAV-SDN

- Extraction of host details like IP, MAC address and connected OFSwitch port ID & OFSwitch ID from ARP packets.
- Managing OFSwitch ID & OFSwitch Port ID of a connected host along with host's IP & MAC addresses in a HostTable at SDN controller level.
- Setting up necessary flow entries into flow table of OFSwitch and validating source host IP Address of packets.

3.2 Architecture of SIPAV-SDN

Proposed SIPAV-SDN architecture has various network components like hosts, OFSwitches, SDN Controller. Hosts are connected with OFSwitch through a switch port and OFSwitches are connected with controller and have interconnection links between forwarding devices (Switches, routers, hubs). Architectural diagram of SIPAV-SDN is shown in figure 1. We used RYU Controller to connect with OFSwitches through Open Southbound API. On top of RYU Controller we proposed our SIPAV-SDN application approach. SIPAV-SDN has various modules with specific task and details are described in following sub-section.

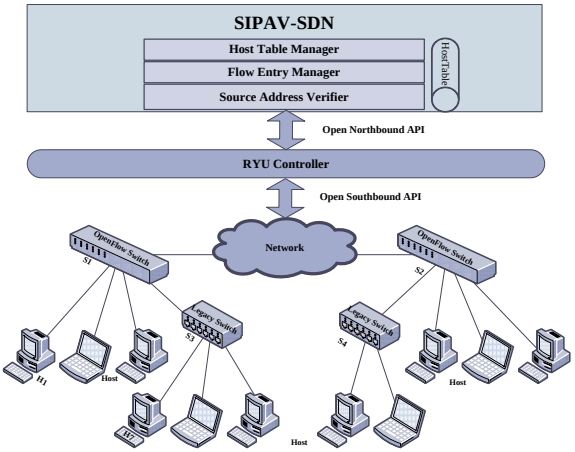


Figure 1: Architecture of SIPAV-SDN

3.3. Modules of SIPAV-SDN

3.3.1. Host Table Manager:

Before starting transmission of data packets every host needs MAC address of target host and it generates ARP-request packets in network with target host IP & broadcast MAC. The target host replies with ARP reply packet to ARP-request sender host. This module is proposed to extract host details like host IP, MAC and connected OFSwitch port ID & OFSwitch ID from ARP packets and to make necessary entry into HostTable. HostTable Manager module checks host detail in HostTable whether exists or not. If a host details do not exist than the details will be added into HostTable. Research paper [29] has discussed very similar type of table to keep track of IP and MAC. The format of our proposed HostTable format is given in figure-2 to store host details.

Switch ID	Switch Port ID	Host MAC	Host IP

Figure-2: HostTable format

After making necessary entries for host details into HostTable, the module sends host details to Flow Entry Manager for further action.

3.3.2. Flow Entry Manager:

This module is designed to create and manage flow entries into flow table of OFSwitch as and when a new host details detected, updated into HostTable and forwarded to Flow Entry Manager by Host Table Manager. This module creates flow entries into flow table for forwarding of host ARP & IP packets to its destination.

First, this module setups all required flow entries for IP packets into flow table of OFSwitch using source and target host fields for matching MAC, IP address, Switch Port ID.

Secondly, it setups required flow entries for ARP packets into flow table of OFSwitch using source and target host fields for matching MAC, Switch Port ID.

Finally, it installs flow entries for ARP packets having broadcast address as destination and assigns output ports as controller port number so such packets get forwarded to controller for inspection and decision as per security policy.

The host packets not matching with flow entries of flow table of OFSwitch are forwarded to controller for further inspection and decision as per security policy.

3.3.3. Source Address Verifier:

This module is performing source IP address validation at controller level by verifying source host details of packet with HostTable entries. If source host details are verified successfully than packet will be forwarded to next hop/destination port otherwise packet will be dropped by this module.

IV. SIPAV-SDN IMPLEMENTATION

Entries in flow table of forwarding device (OFSwitch) are carried out by controller by generating control packets. Whenever a packet is transmitted by host OFSwitch matches flow table entries with packet field values and if packet matches any flow table entry then packet is transmitted for output/destination port otherwise packet is forwarded to controller for further decision. Than controller inspects packet as per security policy, if packet is not as per policy it will be dropped by controller. If packet is as per policy, controller makes flow entries into flow table of OFSwitch. Than device will be enable to directly forward the same type of packet received from the host to destination port. In this case, packets forwarded to controller for policy decision are also sent to destination port of the OFSwitch. Algorithm & flow chart for SIPAV-SDN approach are as given below respectively Algorithm 1 and figure 3:

Algorithm 1: SIPAV-SDN

Implementation at: Packet_In event
Input : env packet_in event massage
Output: Host_details for HostTable

1. Extract DstIP, DstMAC, Switch ID & Source Port ID, PacketType from evn // host detail extraction
2. If PacketType = IP then goto 15
3. If SrcIP, SrcMAC not exists in HostTable than //verification of source host IP and MAC with HostTable
4. For x = 1 to HostTable (Switch ID) do // switch ID searching in HostTable
5. FlowRule = Source_MAC= SrcMAC,
   In_port = Switch Port ID
   Dst_MAC = (HostTable(x).DstMAC)
   Out_port = (HostTable(x).Switch Port ID)
   Traffic_type = (ARP, IP) // flow entry parameters for ARP and IP type packets
6. Set FlowRule // setup rule into flow table
7. End for
8. FlowRule = Source_IP=SrcIP,
   Source_MAC= SrcMAC,
   In_port = Switch Port ID
   Dst_MAC = (“ff:ff:ff:ff:ff:ff”)
   Out_port = Flood
   Traffic_type = (ARP) // flow entry parameters for ARP type packets
9. Set FlowRule // setup rule into flow table
10. Add SrcIP, SrcMAC, Swich ID, Switch Port ID into HostTable // update host details into HostTable
11. End if
12. Goto 26
13. If Switch ID and Switch Port ID exist in HostTable than // verification of switch ID and port ID with HostTable
14. If SrcIP, SrcMAC exists in HostTable than // verification of source host IP and MAC with HostTable
15. FlowRule = Source_MAC= SrcMAC,
    In_port = out_port
    Out_port = Switch Port ID // flow entry parameters for interconnections between switch
16. Set FlowRule // setup rule into flow table
17. Else
18. Generate alert, drop packet and return
19. End if
20. Else
21. FlowRule = Source_MAC= DstMAC,
    In_Port = out_port
    Out_Port = in_port // flow entry parameters for interconnections between switch
22. Set FlowRule // setup rule into flow table
23. FlowRule = Source_MAC= SrcMAC,
    Source_IP=SrcIP,
    In_port = in_port
    Out_port = out_port
    Traffic_type = (ARP, IP) // flow entry parameters for ARP and IP type packets
24. Set FlowRule // setup rule into flow table
25. End if
26. Forward packet to out_port
27. End

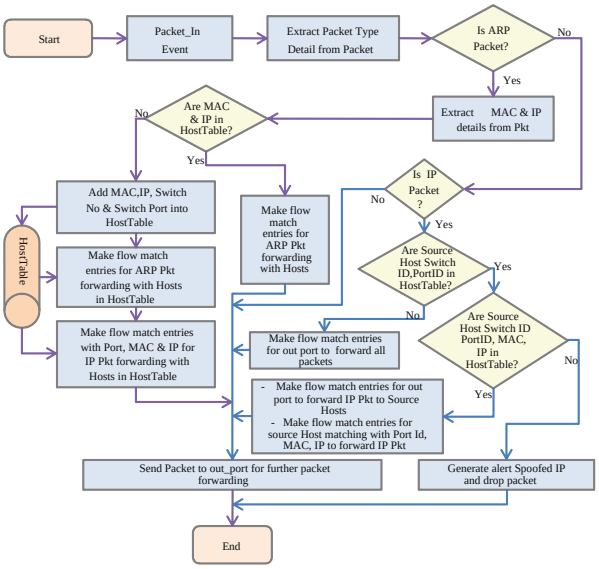


Figure 3: Flow Chart for SIPAV-SDN

The proposed approach is implemented at PacketIn event of the controller. OFSwitch communicates all packets to controller through PacketIn event. This event is responsible for inspection of packet, implementation of packet security & forwarding policies and creation of flow table entries in the connected OpenFlow switches. Approach used ARP packets for collection of host information and store into HostTable.

The HostTable is having fields like Switch ID, Switch Port ID & MAC and IP address of connected host to the switch port. These field values are used for making flow entries into flow table and for verification of source IP address at controller level.

V. SIPAV-SDN PERFORMANCE

In this part, we have evaluated performance of proposed system based on number of flow entries generated, bandwidth test with SIPAV-SDN and without SIPAV-SDN and Flow Entry Generation Comparison among SDN-SAVSH, SDN-SAVI and SIPAV-SDN. The objectives of this paper are extraction of host details like IP, MAC, Switch ID & Switch Port ID, management of HostTable at controller level, setup required flow entries into flow table of OFSwitch table and verify packet's source host IP address. Proposed system setups required flow entries into flow table for forwarding of subsequent same type of packet directly to its destination by matching with flow entries in its flow table.

5.1. Simulation Environment:

The proposed system implemented and results generated for fulfillment of its objectives. The experiment was conducted using an Intel® Core(TM) i5-6200U CPU @2.30Ghz 2.40Ghz with 4 GB RAM and 500GB SSD Disk, installing and running Ubuntu 18.04 LTS. SDN Controller RYU-4.28 with python 2.7.12 support was configured for emulation of various SDN network topologies. We used Wireshark-2.2.6 to capture and inspect packets travelling between various switches and hosts. Gedit text editor and Scapy software library were used to generate Python programs as per experiment requirements.

5.2. SDN Topology Scenarios:

During simulation of system, we used various OpenFlow Network topologies for implementation and experimentation of SIPAV-SDN. There we designed a Single OpenFlow Switch Topology (SOST) with Single OpenFlow Switch, 4 hosts and an SDN controller, a Multiple OpenFlow Switch Topology (MOST) with 3 OpenFlow switches, 7 hosts and an SDN Controller and a Hybrid OpenFlow Switch Topology (HOST) with 2 OpenFlow Switches, 2 legacy switches, 11 hosts and an SDN controller. The diagram of above mentioned various topology configurations are shown below in figure 4, 5 & 6:

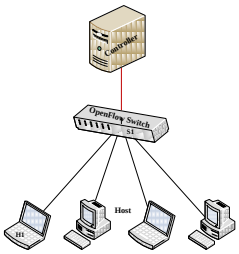


Figure 4: Single OpenFlow Switch Topology (SOST)

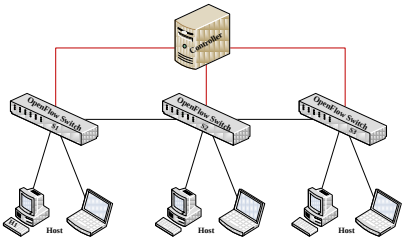


Figure 5: Multiple OpenFlow Switch Topology (MOST)

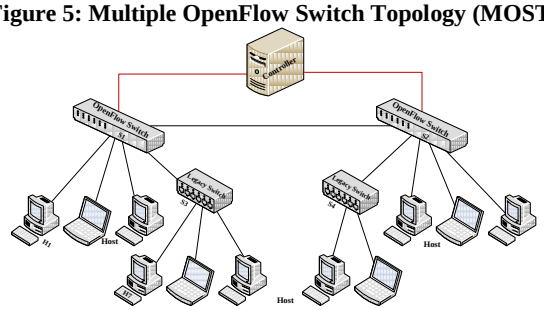


Figure 6: Hybrid OpenFlow Switch Topology (HOST)

5.3. Topology-wise HostTable Record:

The proposed system used SOFT, MOST & HOST network topologies during experimentation and it detected connected host in network topology and extracted host details using ARP packets generated by hosts. The extracted details of connected hosts have been stored in the HostTable as shown in table 1, 2 & 3. We added new hosts manually during experiment and system entered newly added host details into HostTable. The changes made during experimentation also shown in table 4 in bold and italicized font face. These results have been verified with Wireshark network packet analyzer. It shows that proposed system worked properly and detected the connected hosts successfully.

Table 1: HostTable record for Topology HOST

Switch ID	Switch Port ID	Host MAC	Host IP
2	3	00:00:00:00:00:06	10.0.0.6
	2	00:00:00:00:00:05	10.0.0.5
	1	00:00:00:00:00:04	10.0.0.4
	4	00:00:00:00:00:11	10.0.0.11
		00:00:00:00:00:10	10.0.0.10
1	1	00:00:00:00:00:03	10.0.0.3
	5	00:00:00:00:00:02	10.0.0.2
	4	00:00:00:00:00:01	10.0.0.1
	2	00:00:00:00:00:08	10.0.0.8
		00:00:00:00:00:07	10.0.0.7
		00:00:00:00:00:09	10.0.0.9

Table 2: HostTable record for Topology MOST

Switch ID	Switch Port ID	Host MAC	Host IP
2	2	00:00:00:00:00:03	10.0.0.3
	3	00:00:00:00:00:04	10.0.0.4
1	2	00:00:00:00:00:02	10.0.0.2
	3	00:00:00:00:00:01	10.0.0.1
3	2	00:00:00:00:00:06	10.0.0.6
	1	00:00:00:00:00:05	10.0.0.5

Table 3: HostTable record for Topology SOST

Switch ID	Switch Port ID	Host MAC	Host IP
1	2	00:00:00:00:00:01	10.0.0.1
	1	00:00:00:00:00:02	10.0.0.2
	3	00:00:00:00:00:03	10.0.0.3
	4	00:00:00:00:00:04	10.0.0.4

Table 4: HostTable record for Topology SOST with Different/Multiple IPs but same MAC

Switch ID	Switch Port ID	Host MAC	Host IP
1	2	00:00:00:00:00:02	10.0.0.2, 10.0.0.12, 10.0.0.13
	3	00:00:00:00:00:03	10.0.0.3
	4	00:00:00:00:00:04	10.0.0.4
	5	00:00:00:00:00:05	10.0.0.5
	6	00:00:00:00:00:06	10.0.0.6
	7	***00:00:00:00:00:07***	***10.0.0.7***
	9	***00:00:00:00:00:09***	***10.0.0.9, 10.0.0.11***
	8	***00:00:00:00:00:08***	***10.0.0.8***
	10	***00:00:00:00:00:0a***	***10.0.0.10***
	1	00:00:00:00:00:01	10.0.0.1

5.4. Topology Scenario-wise Number of Flow Entries:

SOST, MOST & HOST network topologies having different configuration such as number of connected hosts, OFSwitches and others were used for proposed system implementation for extraction of host details and setting up flow entries into flow table of connected OFSwitches of network. Details of flow entries made during experimentation have been read using **ovs-ofclt dump-flows** command and number of flow entries were recorded in table 5. It shows that system framed and installed required flow entries into flow table of OFSwitches.

Table 5: Flow Entries generated during implementation of approaches in various topologies

Topology Scenario	No of flow Entries			
	OpenFlow Switch			Total
	S1	S2	S3	
SOST	31	-	-	31
MOST	26	33	27	86
HOST	131	111	-	242

5.4. Varying in Number of Hosts to Evaluate Flow Entries Generation:

We conducted experiments with SOST topology by varying the number of hosts and checked flow entries into flow table of connected OFSwitch. The numbers of flow entries in flow table of *S1* OFSwitch with varying number of hosts during experiment were show in table 6. It shows that system worked properly with varying number of hosts in SDN network and framed and installed required flow entries into flow table of OFSwitch.

Table 6: Implementation with varying No of Hosts

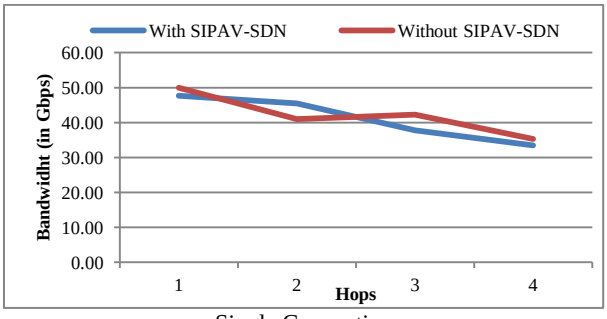
Test Sequence No	Hosts	Flow Entries
1	15	449
2	30	1799
3	45	4049
4	60	7199
5	75	11249
6	90	16199
7	105	22049
8	120	28799
9	135	36499
10	150	44999

5.5. Bandwidth Test with and without SIPAV-SDN at various Hops:

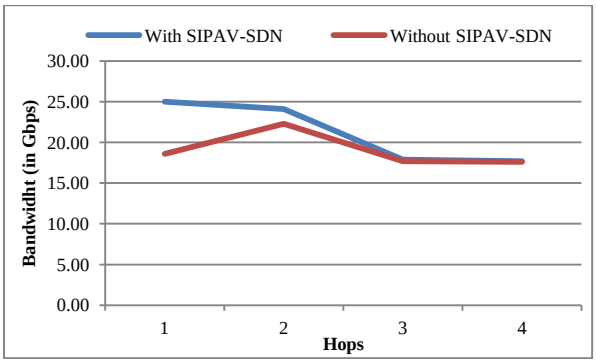
Bandwidth test was conducted with HOST topology by varying the number of hops with single, dual & multiple connections. Bandwidth was tested with & without SIPAV-SDN and results are shown in table 7 and graphical representation in figure 7.

Table 7: Bandwidth Performance

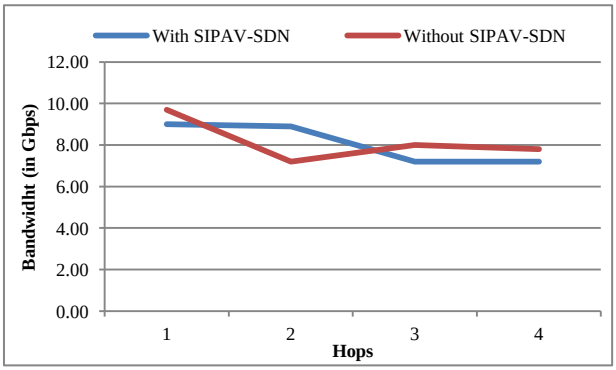
Connections		Hops (Bandwidth in Gbps)			
		1	2	3	4
With SIPAV-SDN	Single	47.70	45.50	37.80	33.50
	Dual	25.00	24.10	17.90	17.70
	Multiple	9.00	8.90	7.20	7.20
Without SIPAV-SDN	Single	50.00	41.00	42.30	35.30
	Dual	18.60	22.30	17.70	17.60
	Multiple	9.70	7.20	8.00	7.80



Single Connection



Dual Connections



Multiple Connections

Figure 7: Bandwidth Performance.

5.6. Comparison among SDN-SAVSH, SDN-SAVI and SIPAV-SDN:

Comparison for generation of flow entries has been carried among SIPAV-SDN, SDN-SAVSH & SDN-SAVI. We have considered results of SIPAV-SDN shown in table 6 for SOST topology recorded during experimentation. Other approaches such as SDN-SAVSH and SDN-SAVI have been studied and taken results from [8] and calculated flow entry generation ratio as shown in table 8 & figure 8. Initially SIPAV-SDN generated more flows then other approaches but after deployment ratio 0.70, it generated flow entries below SDN-SAVSH and above SDN-SAVI.

Table 8: Flow Entry Generation Comparison of among SDN-SAVSH, SDN-SAVI and SIPAV-SDN

Deployment of SIPAV-SDN		Flow Entry Generation Ratio		
No of Hosts	Flow Rules Generated	SIPAV-SDN	SDN-SAVSH*	SDN-SAVI*
15	449	0.50	0.30	0.10
30	1799	0.67	0.50	0.20
45	4049	0.75	0.60	0.30
60	7199	0.80	0.66	0.40
75	11249	0.83	0.70	0.50
90	16199	0.86	0.75	0.60
105	22049	0.87	0.88	0.70
120	28799	0.89	1.00	0.80
135	36499	0.92	1.00	0.90
150	44999	1.00	1.00	1.00
* Data taken from research paper [8]				

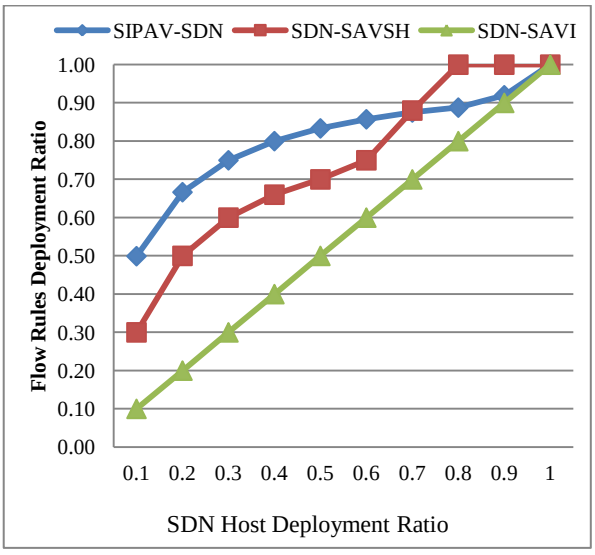


Figure 8: Comparison of among SDN-SAVSH, SDN-SAVI and SIPAV-SDN.

5.7. Performance of SIPAV-SDN during various IP spoofing Attacks:

During experiment we observed results shown in table 9. We used host H1 as attacker which was connected with SDN switch S1 of various SDN topologies. Results have shown that performance of SIPAV-SDN for filtering of IP spoofed packets for attacks of TCP, UDP and ICMP is 100%. This accuracy level was also achieved with varying number of Hosts in SOST topology in between 15 to 150. Performance of source IP spoofed packet filtering for ARP packet was observed between 8% and 20% during experimentation. In other test, we used host H7 as attacker which was connected with legacy (Non SDN) switch S3 of HOST topologies. Result depicted that an attacker host connected with a legacy switch, can successfully carry out an IP spoofing attack on hosts connected with the same switch. This type of attack will not be determined and filtered by an SDN Source Address Validation System if target host and attacker host are connected with the same legacy switch but IP spoofed packets will be detected and filtered incase target or attacker or both are connected with SDN Switch.

:

Table 9: Response of Various IP Spoofing Attacks

Attack Type	Attacker Host Generated Packets		System Identified Packets		Filtering Accuracy Percentage
	Spoofed	Valid	Spoofed	Valid	
Network Topology : HOST					
TCP	1080	90	1080	90	100.00
UDP	1080	90	1080	90	100.00
ICMP	1080	90	1080	90	100.00
ARP	1080	90	90	1080	8.33
Network Topology : MOST					
TCP	315	45	315	45	100.00
UDP	315	45	315	45	100.00
ICMP	315	45	315	45	100.00
ARP	315	45	45	315	14.29
Network Topology : SOST					
TCP	135	27	135	27	100.00
UDP	135	27	135	27	100.00
ICMP	135	27	135	27	100.00
ARP	135	27	27	135	20.00

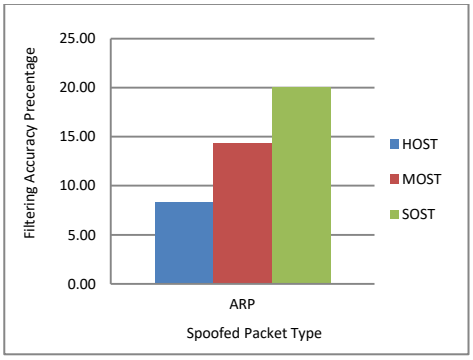


Figure 9: ARP IP spoofed packet Filtering Accuracy in various topologies.

In this test, we used host h7 as attacker which was connected with legacy switch S3 of HOST topologies:

Table 10: IP Spoofing Attacks from Non-SDN Switch connected host

Attack Type	Attacker Host Generated Packets		System Identified Packets		Filtering Accuracy Percentage
	Spoofed	Valid	Spoofed	Valid	
Network Topology : HOST					
TCP	1080	90	864	306	80.00
UDP	1080	90	864	306	80.00
ICMP	1080	90	864	306	80.00
ARP	1080	90	72	1098	6.67

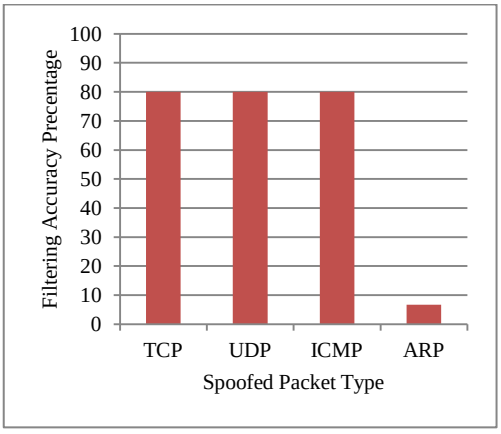


Figure 19: Filtering Accuracy for Non-SDN Switch connected host.

VI. SIPAV-SDN FEATURES & FUTURE WORKS

Proposed approach is having various features such as it is suitable for small & large SDN network, provides scalability in network, auto-configurable of flow entries in flow table of OpenFlow switches as per security policy. It suggests any change or implementation at controller level and not host level and packet header level. Proposed solution also worked with hybrid networks. During system performance test, we found that processing time for first packet of a host took more time in inspection, forming control packets at controller level and forwarding packet to its destination. So, a study is needed to reduce the processing time of first packet of a host. Other issued is that we used ARP packets for extraction of host details and installation of flow entries into flow table of an OFSwitch. Attacker sends IP spoofed ARP packets than proposed system doesn't perform well due to extraction of host details from ARP packets. So there, is need of a special host discovery system which removed the dependency of ARP packet generated by connected hosts.

VII. CONCLUSIONS

This paper described SIPAV-SDN a new approach for validation of Source Host IP Address for SDN environment by binding source host IP & MAC with switch port. Approach tested using various SDN topologies and stored details such as IP, MAC, connected switch ID and Switch Port ID into HostTable. Test results have shown that filtering accuracy of SIPAV-SDN for IP spoofed TCP, UDP and ICMP packets was 100% and this accuracy level was also achieved with varying number of Hosts for SOST topology in between 15 to 150. Accuracy for source IP spoofed packet filtering for ARP packet was observed low and it was between 8% and 20%. SIPAV-SDN has detected all IP spoofed packets and filtered incase target or attacker or both are connected with SDN Switch but attacker host connected with a legacy switch, can successfully carry out an IP spoofing attack on hosts connected with the same switch.

REFERENCES

1. Akhunzada, E. Ahmed, A. Gani, M. K. Khan, M. Imran and S. Guizani, "Securing software defined networks: taxonomy, requirements, and open issues," in IEEE Communications Magazine, vol. 53, no. 4, pp. 36-44, April 2015.
2. Prakash and R. Priyadarshini, "An Intelligent Software defined Network Controller for preventing Distributed Denial of Service Attack," 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, 2018, pp. 585-589.
3. H. Lawal and A. T. Nuray, "Real-time detection and mitigation of distributed denial of service (DDoS) attacks in software defined networking (SDN)," 2018 26th Signal Processing and Communications Applications Conference (SIU), Izmir, 2018, pp. 1-4.
4. Zhang, G. Hu, G. Chen, A. K. Sangaiah, P. Zhang, X. Yan and W. Jiang, "Towards a SDN-Based Integrated Architecture for Mitigating IP Spoofing Attack," in IEEE Access, vol. 6, pp. 22764-22777, 2018
5. Satasiya and Raviya Rupal D., "Analysis of Software Defined Network firewall (SDF)," 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), Chennai, 2016, pp. 228-231
6. Nordmark, M. Bagnulo, E. Levy-Abegnoli "FCFS SAVI: First-Come, First-Served Source Address Validation Improvement for Locally Assigned IPv6 Addresses", Internet Engineering Task Force, RFC-6620 ISSN:2070-1721 May 2012
7. Gian M. di Marzo and Francesco Benedetto, "Software Defined Networks for Data Center Optimization", Recent Patents on Computer Science (2014) 7: 24.
8. Guolong Chen, Guangwu Hu, Yong Jiang and Chaoqin Zhang, "SAVSH: IP source address validation for SDN hybrid networks," 2016 IEEE Symposium on Computers and Communication (ISCC), Messina, 2016, pp. 409-414
9. H. T. Nguyen Tri and K. Kim, "Assessing the impact of resource attack in Software Defined Network," 2015 International Conference on Information Networking (ICOIN), Cambodia, 2015, pp. 420-425
10. K. Guerra Pérez, X. Yang, S. Scott-Hayward and S. Sezer, "A configurable packet classification architecture for Software-Defined Networking," 2014 27th IEEE International System-on-Chip Conference (SOCC), Las Vegas, NV, 2014, pp. 353-358.
11. K. K. Karmakar, V. Varadharajan and U. Tupakula, "Mitigating attacks in Software Defined Network (SDN)," 2017 Fourth International Conference on Software Defined Systems (SDS), Valencia, 2017, pp. 112-117
12. Kwon, Jonghoon, Dongwon Seo, Minjin Kwon, Heejo Lee, Adrian Perrig and Hyogon Kim. "An incrementally deployable anti-spoofing mechanism for software-defined networks." 2015 Computer Communications 64: pp.1-20.
13. L. M. van Adrichem, Niels & Doerr, Christian & A. Kuipers, Fernando. (2014). "OpenNetMon: Network monitoring in OpenFlow Software-Defined Networks." IEEE/IFIP Network Operations and Management Symposium: Management in a Software Defined World 10.1109/NOMS.2014.6838228: pp. 1-8.
14. M. Dabbagh, B. Hamdaoui, M. Guizani and A. Rayes, "Software-defined networking security: pros and cons," in IEEE Communications Magazine, vol. 53, no. 6, pp. 73-79, June 2015.
15. M. Z. Masoud, Y. Jaradat and I. Jannoud, "On preventing ARP poisoning attack utilizing Software Defined Network (SDN) paradigm," 2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), Amman, 2015, pp. 1-5.
16. Manzanares-Lopez, Pilar & Muñoz-Gea, Juan & Manuel Delicado-Martinez, Francisco & Malgosa, Josemaria & Flores de la Cruz, Adrian. (2016). Host Discovery Solution: An Enhancement of Topology Discovery in OpenFlow based SDN Networks. 80-88.
17. Michele Amoretti, Gianluigi Ferrari, Jean-Luc Richier and Andrzej Duda, "Patents on IPv6-Related Technologies", Recent Patents on Computer Science (2013) 6: 170.
18. O. Chippalkatti and S. U. Nimbhorkar, "An approach for detection of attacks in software defined networks," 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, 2017, pp. 1-3.
19. O. Strugaru, A. D. Potorac and A. Graur, "The impact of using Source Address Validation filtering on processing resources," 2014 10th International Conference on Communications (COMM), Bucharest, 2014, pp. 1-4
20. P. B. Pawar and K. Kataoka, "Segmented proactive flow rule injection for service chaining using SDN," 2016 IEEE NetSoft Conference and Workshops (NetSoft), Seoul, 2016, pp. 38-42
21. P. Zanna, S. Hosseini, P. Radcliffe and B. O'Neill, "The challenges of deploying a software defined network," 2014 Australasian Telecommunication Networks and Applications Conference (ATNAC), Southbank, VIC, 2014, pp. 111-116.

22. Ramesh Chand Meena, Mahesh Bundele, "A Review on Implementation Issues in IPv6 Network Technology", International Journal of Engineering Research and General Science(2015), Vol.3, Issue 6: pp.800-809
23. S. Murtuza and K. Asawa, "Mitigation and Detection of DDoS Attacks in Software Defined Networks," 2018 Eleventh International Conference on Contemporary Computing (IC3), Noida, 2018, pp. 1-3.
24. S. Nadar and S. Chaudhari, "Proactive-routing path update in Software Defined Networks(SDN)," 2017 International Conference on Intelligent Computing and Control (I2C2), Coimbatore, 2017, pp. 1-3.
25. S. T. Ali, V. Sivaraman, A. Radford and S. Jha, "A Survey of Securing Networks Using Software Defined Networking," in IEEE Transactions on Reliability, vol. 64, no. 3, pp. 1086-1097, Sept. 2015.
26. S.K. Agrawal and Kapil Sharma*, "Millimeter Wave Channel Capacity for 5th Generation Software Defined Radio Communication System in Vegetation Area", International Journal of Sensors, Wireless Communications and Control (2018) 8: 172.
27. T. Alharbi, M. Portmann and F. Pakzad, "The (in)security of Topology Discovery in Software Defined Networks," 2015 IEEE 40th Conference on Local Computer Networks (LCN), Clearwater Beach, FL, 2015, pp. 502-505.
28. T. Chin, X. Mountrouidou, X. Li and K. Xiong, "Selective Packet Inspection to Detect DoS Flooding Using Software Defined Networking (SDN)," 2015 IEEE 35th International Conference on Distributed Computing Systems Workshops, Columbus, OH, 2015, pp. 95-99
29. T. Javid, T. Riaz and A. Rasheed, "A layer2 firewall for software defined network," 2014 Conference on Information Assurance and Cyber Security (CIACS), Rawalpindi, 2014, pp. 39-42.
30. T. Xu, D. Gao, P. Dong, C. H. Foh and H. Zhang, "Mitigating the Table-Overflow Attack in Software-Defined Networking," in IEEE Transactions on Network and Service Management, vol. 14, no. 4, pp. 1086-1097, Dec. 2017.
31. Y. Jia, Y. Liu, G. Ren and L. He, "Revisiting inter-AS IP spoofing let the protection drive source address validation," 2017 IEEE 36th International Performance Computing and Communications Conference (IPCCC), San Diego, CA, 2017, pp. 1-10.
32. Bingyang Liu, Jun Bi and Yu Zhou, "Source Address Validation in Software Defined Networks," in SIGCOMM'16 August 22-26 2016, p. 595.
33. Alif Akbar Pranata, Tae Soo Jun, Dong Seong Kim, "Overhead reduction scheme for SDN-based Data Center Networks," Computer Standards & Interfaces, Volume 63, 2019, Pages 1-15, ISSN 0920-5489,
34. Mingyong Chen, Weimin Wu, "A first packet processing subdomain cluster model based on SDN", AIP Conference Proceedings-1864, 020042 (2017)
35. D. Kotani and Y. Okabe, "Packet-In Message Control for Reducing CPU Load and Control Traffic in OpenFlow Switches," 2012 European Workshop on Software Defined Networking, Darmstadt, 2012, pp. 42-47. doi: 10.1109/EWSDN.2012.23
36. S. Iyer, V. Mann and N. R. Samineni, "SwitchReduce: Reducing switch state and controller involvement in OpenFlow networks," 2013 IFIP Networking Conference, Brooklyn, NY, 2013, pp. 1-9.
37. Patel, Sonal and Vikas Jha. "Various Anti IP Spoofing Techniques." Journal of Engineering Computers & Applied Sciences(JECAS) Journal of Engineering Computers & Applied Sciences(JECAS) Volume 4, No.1, January 2015 pp 27-31

AUTHORS PROFILE

Ramesh Chand Meena is currently pursuing the Ph.D. degree in Computer Science & Engineering with the Poornima University, Jaipur, India. He is also working as Scientist with Software Technology Parks of India, Ministry of Electronics and IT, Government of India. He completed Master of Technology in Computer Science and Engineering in 2006, Master of Science in Computer Sciences in 2003. He has more than 20 years of working experience at various organizations and departments of Government of India in the field of Computer and IT. His research interests include Network Architecture and Security, Computer Programming, Software Defined Networks etc.

Dr. Meenakshi Nawal is currently working as Associate Professor with the Poornima University, Jaipur, India. She is Gold Medalist in M.Sc (IT) from MDS University Ajmer. She has completed Ph. D (Computer Science) from Banasthali Vidhyapith, Jaipur. She is having 13 years experience of Academics and Research. Her area of research is "Patient Authentication and Security measures in Remote Health Monitoring". She has attended many National and International Workshops, Conferences and Published papers in National conference. Her areas of research interest are Machine Learning, Deep Learning, Image Processing, Big Data Analytics and Software Defined Networks etc.

Professor (Dr.) Mahesh M. Bundele has completed his Bachelor's degree in Electronics and Power in 1986 from Nagpur University and immediately joined as Lecturer in Electronics at Babasaheb Naik College of Engineering, Pusad, Yavatmal district. He did his Master's in Electrical Power System and Doctoral in Computer Science & Engineering with a topic "Design and Implementation of Wearable Computing System for the Prevention of Road Accidents" from Amravati University in 1990 and 2013 respectively. He has worked as Lecturer, Assistant Professor, Professor and Head of CSE & IT during 25 Years at BNCOE and guided many research projects at UG and PG level on various applications in Electrical, Electronics & Computer Sciences including city and village Wi-Fi/Wi-Max design up to 195 villages. He has worked on various govt. and industry research projects. He was also appointed as Principal of Babsaheb Naik College of Engineering, Pusad from March 2011. He has worked in various capacities such as, member Board of Studies, Chief-Valuation officer etc. at University level. He has also worked for getting ISI to Krishak Motor pumps at Amravati. He has visited US, UK, China and Malaysia for research presentations. He has published more than 50 research papers in National and International conferences and journals. He is senior member of IEEE, Life member of ISTE and IEI and the member of ACM. Presently he is working as Member, STDCOM Technical & Professional Activities, and Member Execom, IEEE Delhi Section & Secretary, IEEE Rajasthan Subsection. He is having total 34 years of teaching including 7 years of research and internship. Presently he is working as Principal & Director, Poornima College Engineering, Jaipur, India. He has also worked Dean (R&D) at Poornima University and Heading Advanced Studies & Research Center dealing Master of Technology programs and Doctoral degree programs of the University. His areas of research interest are Wearable & Pervasive Computing, Software Defined Networks, Wireless Sensor Networks, and Smart Grid Issues etc.