International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-4, November 2019

Improving Security through Software Defined Networking (SDN): AN SDN based Model

Pradeep Kumar Sharma, S. S. Tyagi

Abstract: In this era of virtualization & cloud computing, software defined networking has created a lots of buzz and has shown a great innovation through network programmability towards the better future networks. In this paper the rich concept of the SDN has been compared with the traditional networks based on the fundamental models. This paper evaluates the current security status, and states what can be done to improve it through SDN. In this work an SDN based security approach has been taken into consideration with advanced analysis on security aspects which shows how the overall security of the network can be enhanced with SDN. Mininet which is an SDN emulator has been utilized to perform various operations for implementation, testing and analysis. The proposed Security model based on SDN analytics has been presented with implementation and promising research directions. This work can be beneficial for researches as well as network engineers in the area of SDN who want to secure and scale their network through the flexibility in SDN

Keywords: Control plane, data plane, openflow, SDN,

I. INTRODUCTION

The fundamentals of the network and system administration have been changed a lot during the last years. The evaluation of virtualization has changed the era of system and server administration by providing an elastic environment through hypervisor. With the advent of operating system virtualization, now a days we are running several severs on a single machine. With further development in memory, processing power, and storage technology, data center hardware can run numerous servers at the same time in a virtual environment. But in the area of networking, we are still using the vendor dependent hardware with their own proprietary software which we have to replace according to the increasing demands and these vendor specific devices has no option for programmability and enhancement. SDN is a concept which targets on all these issues and leads towards better future networks [1].

The contents of the paper are arranged as follows: In section II we discuss about the origin of SDN and its architecture. Section III is based on comparative study on traditional networking model & SDN. Section IV features our approach to security challenges and opportunities of enhancing security in SDN with new ways to threat mitigation; Section V is all about the implementation and future scope. Finally part VI concludes the analysis with research directions.

Revised Manuscript Received on November 15, 2019

Pradeep Kumar Sharma, Assistant Professor, in Computer Engineering at Manav Rachna International Institute of Research and Studies (MRIIRS), Faridabad.

Dr. S. S. Tyagi, Professor, Computer Engineering and Dean at Manav Rachna International Institute of Research and Studies (MRIIRS), Faridabad.

II. ORIGIN OF SDN AND ARCHITECTURE

The work towards isolation of control logic from data logic has a long history. But it came in limelight in 2006, when Martin Casado, a PhD understudy at Stanford University and group propose a new security design (SANE) which characterizes a unified control of security (rather than at the edge as typically done). It states that security should be checked at each entrance as well as main entrance in the network. Ethane sums it up to all arrangements providing ethane switches to provide a hybrid network environment as it was not possible to replace the whole existing network [2]. The possibility of Software Defined Network happened from OpenFlow venture (ACM SIGCOMM 2008) [3]. In 2009 Stanford announced OpenFlow V1.0.0 specs and Martin Casado again helped and established Nicira in June 2009. In March 2011 Open Networking Foundation was framed and First Open Networking Summit was hung on Oct 2011. Numerous Industries Juniper, Cisco declared to consolidate. In July 2012 VMware purchases Nicira for $1.26B.

SDN is based on the concept of data plane and control plane. A network can be viewed as constitute of data and control plane. The data plane is responsible for forwarding the data as per the flow rules and control plane defines the flow rules and control decisions necessary for the delivery of user data to right destination. In traditional networking this all comprises in a single box (e.g. Routers). In SDN the controlling part of the network has been decoupled from the inter- networking devices to a logically centralized controller and these network devices work as the general purpose data forwarding devices.For clarity, SDN is described in this article with the Open Networking Foundation (ONF)[4] definition: "In the SDN architecture, the control and data planes are decoupled, network intelligence and state are logically centralized, and the underlying network infrastructure is abstracted from the applications."

SDN focuses on four key features:

- Isolation of logical intelligence from the devices
- A central place for all intelligence and control
- APIs between the data logic and control logic i.e. controller and devices
- Innovation through programmability
- Increased Security and reliabilities with complete visibility and control over the network.

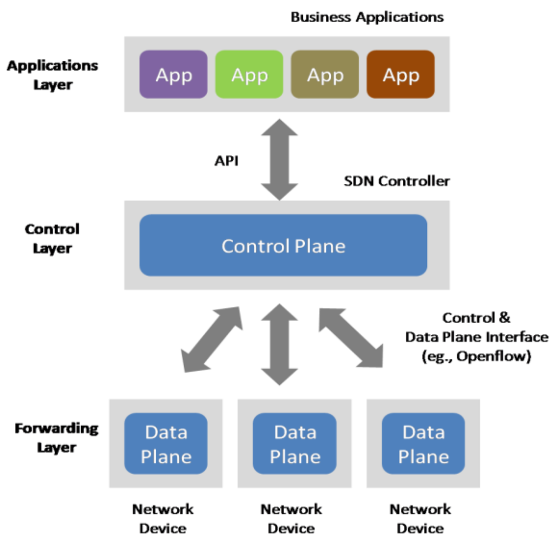


Figure 1. Architecure of SDN [5]

Figure 1 shows the basic architecture of the SDN. The basic working of the SDN includes the communication of controller with data plane. The controller does this by using the open flow protocol. Open flow protocol works as a communication medium between the controller and forwarding devices and encourages the decoupling of control from the network devices. This is a flow based communication; each device in the data plane maintains a flow table which is managed by the controller. To maintain the communication over the network, an open flow controller adds and removes the forwarding rules in network switches. A forwarding rule is based on the match of the fields (packet header, incoming port etc.)e.g. source and destination IP addresses and related actions are performed e.g. forward or drop a packet. To configure a new policy in switch, the controller can modify relevant entries in the flow tables and this may also be done in real time.

III. TRADITIONAL NETWORKING AND SDN

In traditional networking the control plane and data plane resides inside the networking device. Every device (e.g. Routers) has its control plane and takes decisions as per the configured policy/protocol as shown in figure 2 and 3. Once the policies have been configured and flow has been defined it is very difficult to change the network behavior in response to changing traffic demands. The only way to make an adjustment is to change the configuration of all the devices. This leads to a bottleneck for the administrators who want to scale their network as per the demands. With the increase in use of the mobile devices, cloud computing and big data demand a great need of change the network behavior in the real time.

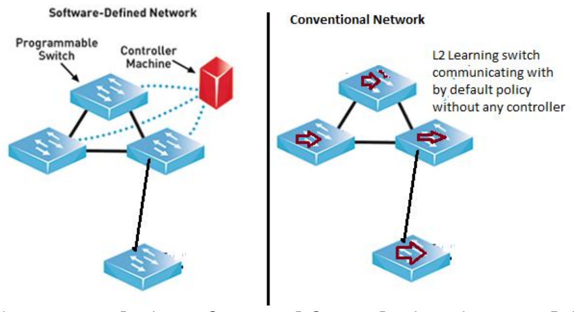


Figure 2.Isolation of control from devices in SDN [5]

Figure 2 shows the conceptual design of SDN where controlling part of a network device has been separated to a logically centralized controller and networking devices are just switches which can work fast and efficiently. Security solutions in traditional networks use a lots complex mechanism to protect the network namely ACLs, VLAN, firewall, NAT etc. These policies are distributed on all the networking devices. The policies are topology based; address based and even port based which breaks as per the changes in network topology or user move.

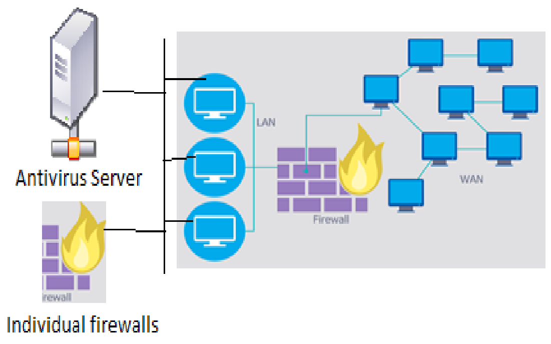


Figure 3. Traditional Security Architecture

A set of all security policies is put in one box i.e firewall and it is kept at the entry and exit point of the network as shown in figure 3. If an attacker makes it through the firewall it has all the access to the network. Distributed firewalls and antivirus are implanted on end users to mitigate this but it exhibits the complexity of the traditional network and placing all the trust in end users. A traditional firewall can only prevent threats to access your computer on internet while most of the viruses or Trojans are received via emails, through file sharing or through direct download of malicious programs. Traditional firewall cannot prevent this. In most of the firewalls packet filtering is done at network layer and transport layer generally. But nowadays there is a requirement of more enhanced version of firewalls which can even work at Application Layer. Some of the firewalls equip with this facility but they all depend on protocol specification related to particular applications. Proxy, IDS and IPS try to prevent the network attacks but traditional network architecture creates bottleneck having control distributed in devices which creates a lots of complexity for policy enforcement in these networks [5].
In SDN architecture, above the controller there is an application plane which introduces the concept of the programmability in the networks. Here we have different applications like traffic monitoring, security which can be directly programmed as per requirements. While in existing systems the network devices are closed boxes where there is no scope of programmability and innovation. The concept of network programmability is one of the prime implicates of the SDN. Until recently most modern network elements (e.g. routers, switches or firewalls) supported a small set of interfaces that were used to communicate with those elements. These typically included a proprietary command line interface (CLI), SNMP, CORBA and most recently NETCONF. Unfortunately none of these languages are able to provide a complete common solution.

They are very static in nature and require a priori data model design and declaration. SDN relies on having multiple managers, agents and controllers, all interacting in symphony of tightly coupled communication which leads to the optimizations and abilities which cannot be obtained by these old interfacing models. In order to realize this new era of communication and interaction, tightly coupled and bidirectional streaming interfaces are needed. Several application friendly interfaces come into consideration including JSON, Google buffers, Thrift and more recently the work in IETF's I2RS (Interface to Routing System).

IV. OPPORTUNITIES FOR SECURITY ENHANCEMENT IN SDN

SDN system-wide complete view of network, programmability through open application programming interfaces, and control of policies through a centralized entity controller provides various ways for security enhancement and threat mitigation. SDN opens up a new platform to create customized security algorithms [6]. SDN supported network proffer a central place for data collection from network devices and new security approaches assumes a centralized data model which was not possible in conventional networks. This is an extreme transformation which has positive ramification for various algorithms related to network monitoring, and firewall methodologies [7]. In this section we will analyze how SDN work with different terminologies like network monitoring, network verification & automation, threat detection and response, which can identify promising future research directions in these networks.

A. Network Monitoring:

Network monitoring is the fundamental part for network security. Actually, suspicious traffic patterns can be found by collecting the real time data from the network and testing it for security breach through various anomaly detection algorithms, For example an attacker can use scanning tools to know the network behavior before doing attack operation. In this case network monitoring becomes more important. Network monitoring in SDN, based on open flow consists of collection of flow based data at controller side which is a natural open flow process in SDN. This can be achieved in two ways. One through the push operation, when a switch tells the controller about the flow that it is expired (FlowRemoved message). Another way is pull operation when controller asks the forwarding devices to know the status of flows through FlowStatisticsRequest and FlowStatisticsReply messages. FlowSense [8] is an example of push operation.

B. Network Verification and Automation:

Manual policy configuration is always the error prone and there should be some automation techniques for configuration verification and consistency. A survey from Gartner points out that, in a passage of year 2010 to 2015, most of network blackouts affecting vital administrations are because of manual configurations and process related, and over half of them coming from policy changes i.e. reconfigurations and updates issues [9]. In SDN when there are more than one controller, several applications and multiple users running concurrently in the same domain, this may lead to inconsistency and policy violation issues. This can cause several network faults like loops, blackholes and access control issues. Moreover in big networks where there are many switches, controllers need to install thousands of flows dealing with many flow tables, controller can install approximately 50000 new flows every second [10] , there should be brisk, efficient approach to guarantee security consistence, adaptation to non-critical failure, and quick failover. The good work around there, FlowChecker [11] is property-based verifier tools that find different misconfiguration inside the network. FlowChecker uses Binary Decisions Diagrams and encodes switch flow-table configuration to create a state machine depicting the flow statistics of OF forwarding devices in the network. NICE [12] is also another error finding tool in SDN configurations. Moreover except these solutions which are used before the network start or application installation, VeriFlow [13] is an on-fly arrangement which check network accuracy in real-time as the network advances progressively. NOX controller also has an inbuilt error checking solution called FORTNOX [14], which identify conflicting flow rules in real-time

C. Improvised Threat Detection

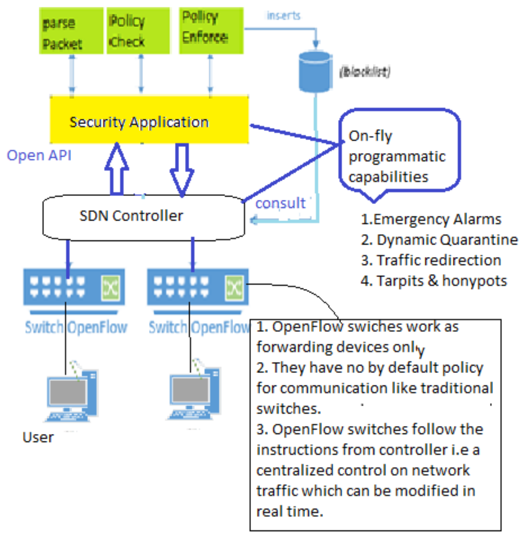
In SDN, controller provides a complete view of the devices which is very much favorable for threat detection. The open flow switches do not have by default communication policy as in L2 learning switches, OF switches follow the instructions from controller and controller can reprogram the data plane devices in the network to conduct analysis for suspicious data and malicious device in the network [15]. Most of the traditional security systems provides security on Layer 3 and layer 4 and cannot detect the malicious payload at application level, in case of application level security in SDN there is need to send all the packets to controllers which create an overhead on controller and respective links. To avoid this situation Mehndi et al [16] proposes an algorithm which is based on the number of unsuccessful connection attempts of fake request. It sends only those packets to the controllers which are suspicious based on the given algorithms. Microsoft is also using SDN solutions in its data centers for malicious traffic detection [17]. With a very large infrastructure of Microsoft conventional packet inspection technology like port mirroring and switch port analyzer (SPAN) are not feasible which require a lots of physical ports and accounting arrangements. In SDN this can be easily configured through controller by using the virtual ports [18]. Radware has used the SDN platform for innovative security solution and provided DefenceFlow for detecting malicious network attacks like DoS [19]. For research and development the open source version of the same has also been provided [20].

D. Dynamic Response to Threats

SDN system-wide complete view of network, programmability through open application programming interfaces, control of policies through a centralized entity controller bolsters the security providers as well researchers and opens up a new ways to provide a dynamic response to threats. Due to the lack of centralized control in legacy network the only response is to drop the malicious traffic but in case of SDN we can redirect the traffic for forensics by reprogram the switches dynamically through the controller. FRESCO [21] and FORTNOX are the example of SDN enabled dynamic response to threats.

V. IMPLEMENTATION, ANALYSIS AND DISCUSSION

Figure 4, shows the implementation scenario of SDN based proposed security application with SDN/Openflow. When a source host send a data packet to destination, the openflow switch check for the matching entry in flow table if a match is found in switch flow table the related action is taken, i.e. the packet is dropped or send to the destination. If no match is found the packet is sent to the controller. The controller sends the packet to the security application policy analysis. The security application first parses the received packet, checks whether the incoming packet violates the security policies or not and enforces a flow rule based on the security policies.
Finally this rule is delivered to switch by the controller and switch update the rule in its flow table. Packet is blocked based on some event associated with an attack signature in the openflow network through Packet_event messages and further packets from this sender blacklisted by the security application. Moreover with the programmability in hand suspicious traffic can also be redirected to a sandbox or quarantine dynamically as per demand.



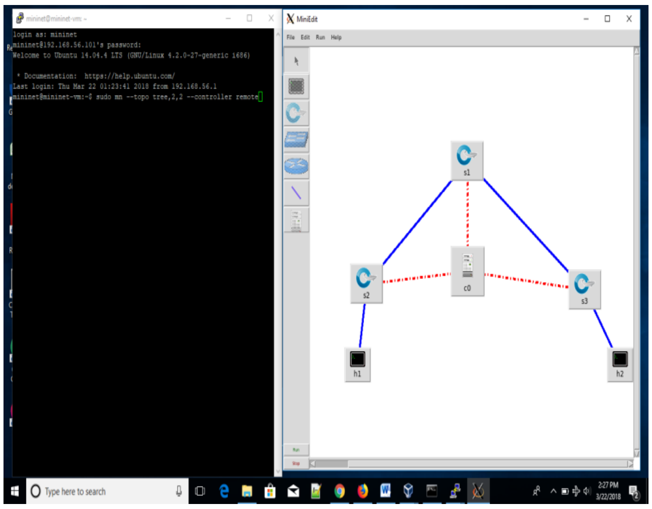
1. OpenFlow swiches work as forwarding devices only
2. They have no by default policy for communication like traditional switches.
3. OpenFlow switches follow the instructions from controller i.e a centralized control on network traffic which can be modified in real time.

Figure 4. SDN based security Architecture

The novelty is:

- Network monitoring and reporting in SDN is more powerful with centralized view and control of the network through the controller.
- Most of the security algorithms supports and work more efficiently on centralized environment as compare to distributed approach which is best fit for network threat detection in SDN.
- With the help of programmability and control, we can generate dynamic responses to the network threats in a more effective way.

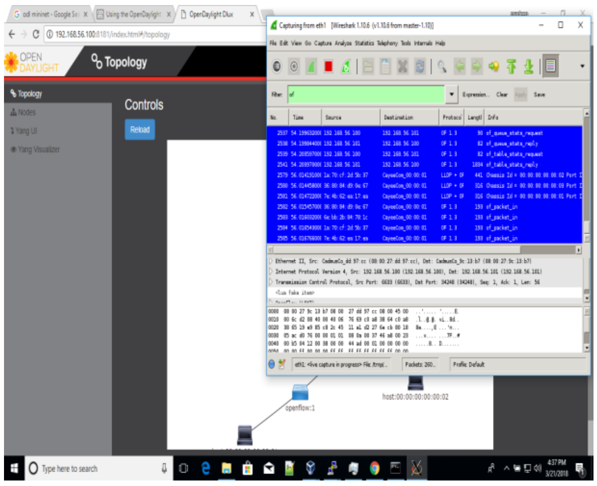
The experimental setup for SDN consists of a controller, open flow switches and hosts as shown in figure 5. For conducting analysis on SDN we are using Mininet. Mininet is SDN network emulator based on Linux. It consist of miniEdit tool which is used for creating the network topology. First the setup is tested for defined topologies with hub code. The hub code is added with the functionalities L2 learning switch. Then setup is tested with openflow supported switch. For simulation purpose there are various tools which are used for analysis of SDN. A virtual image of mininet is provided by github that need to be imported in virtual box. This image does not support graphics so it is needed to use xming server on the host computer. Host computer is used to connect with mininet image. Several network analysis utilities have been used with mininet for conducting the experiments on SDN. For checking the real time traffic patterns wireshark is used. Wireshark is a utility which is used for packet filtering and network analysis in the network. For SDN implementation OpenFlow supported controller POX is used.


```
login as: mininet
mininet@192.168.56.101'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 Mar 22 01:23:41 2018 from 192.168.56.1
mininet@mininet-vm:~$ sudo mn --topo tree,2,2 --controller remote
```

Figure 5. Mininet topology with miniedit

Ovs-ofctl is another tool which is used for seeing the OpenFlow messages and entering the flows in open flow tables. For analyzing the speed for a TCP link iperf utility is used. Cbench is used for testing controllers, deciding the flows and controlling the performance as per flows.
After setting the SDN environment, there is a need to enable security functions. A very basic security functionalty can be easily applied on openVswitch by creating a static firewall blocking traffic from mriu.edu.in –
Table=0, nw_src = 50.28.49.16, actions=drop.
But to design a complete security application we need to understand flow of instructions in SDN. There are two kind of communications in SDN architecture, one is between controller and forwarding devices i.e. southbound API. Another communication is between controller and network applications i.e. Northbound API. This northbound API provides us the functionality to design the various application related to path computation, loop avoidance, routing and security.



```
Capturing from eth1   [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
Filter: of

No.   Time          Source              Destination          Protocol   Length  Info
2537  54.19963200(  192.168.56.100      192.168.56.101       OF 1.3     90      of_queue_stats_request
2538  54.19984400(  192.168.56.101      192.168.56.100       OF 1.3     82      of_queue_stats_reply
2539  54.20559700(  192.168.56.100      192.168.56.101       OF 1.3     82      of_table_stats_request
2541  54.20997000(  192.168.56.101      192.168.56.100       OF 1.3     1894    of_table_stats_reply
2579  56.01419000(  1a:70:cf:2d:5b:37   CayeeCom_00:00:01    LLDP + OF  441     Chassis Id = 00:00:00:00:00:02 Port I
2580  56.01445800(  36:80:84:d9:0e:67   CayeeCom_00:00:01    LLDP + OF  326     Chassis Id = 00:00:00:00:00:03 Port I
2581  56.01472200(  7e:4b:62:ea:17:ea   CayeeCom_00:00:01    LLDP + OF  326     Chassis Id = 00:00:00:00:00:01 Port I
2582  56.01545700(  36:80:84:d9:0e:67   CayeeCom_00:00:01    OF 1.3     193     of_packet_in
2583  56.01603200(  6e:bb:2b:84:78:1c   CayeeCom_00:00:01    OF 1.3     193     of_packet_in
2584  56.01654900(  1a:70:cf:2d:5b:37   CayeeCom_00:00:01    OF 1.3     193     of_packet_in
2585  56.01676600(  7e:4b:62:ea:17:ea   CayeeCom_00:00:01    OF 1.3     193     of_packet_in

Ethernet II, Src: CadmusCo_dd:97:cc (08:00:27:dd:97:cc), Dst: CadmusCo_9c:13:b7 (08:00:27:9c:13:b7)
Internet Protocol Version 4, Src: 192.168.56.100 (192.168.56.100), Dst: 192.168.56.101 (192.168.56.101)
Transmission Control Protocol, Src Port: 6633 (6633), Dst Port: 34248 (34248), Seq: 1, Ack: 1, Len: 56
<Lua fake item>

0000  08 00 27 9c 13 b7 08 00  27 dd 97 cc 08 00 45 00   ..'..... '.....E.
0010  00 6c d2 08 40 00 40 06  76 69 c0 a8 38 64 c0 a8   .l..@.@. vi..8d..
0020  38 65 19 e9 85 c8 2c 45  11 a1 d2 27 6e cb 80 18   8e....,E ...'n...
0030  05 ac d0 76 00 00 01 01  08 0a 00 37 46 a8 00 23   ...v.... ...7F..#
0040  00 b5 04 12 00 38 00 00  44 ad 00 01 00 00 00 00   .....8.. D.......

eth1: <live capture in progress> File: /tmp/...   Packets: 260...   Profile: Default
```

Figure 6. Wireshark with OpenFlow traffic

A variety of northbound API has been created to implement the applications. But till now there is no specific standard for northbound API has been finalized as different applications have different requirements. Based on the various controllers the northbound API can be divided into Representational State Transfer (REST) API, programming languages and other specialized adhoc API that can be used for northbound application development [22]. Floodlight, HP VAN SDN, ONOS, DISCO and OpenContrail are the controllers that use RESTful API as their northbound API. Frenetic, Procera, Nettle NetCore, Pyretic and NetKAT are the example of the northbound API as programming languages.

VI. CONCLUSION

SDN system-wide complete view of network, programmability through open application programming interfaces, control of policies through a centralized entity controller bolsters the security providers as well researchers and opens up a new platform to create a customized security algorithms. In this paper we describe the SDN security aspects with new directions for providing customized on demand security solutions. It has been shows how the SDN can improve overall security by the new ways to find and neutralized the threats. Network monitoring and automation through automatic policy verification in SDN are explained. Implementation plan of SDN driven security approach has also been elaborated with various northbound APIs for application development. Research in software defined networking and northbound API, is still in its early stages, and it provides the great possibilities and opportunities for security enhancement through SDN that weren't previously possible.

REFRENCES

1. M. Casado, T. Garfinkel, A. Akella, M. J. Freedman, D. Boneh, N. McKeown, and S. Shenker, "SANE: A protection architecture for enterprise networks," in Proc. 15th USENIX Security Symp. (SS), 2006, vol. 15.
2 M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "Ethane: Taking control of the enterprise,"ACM SIGCOMM comput. Commun. Rev.(CCR), vol.37, no.4, pp.1–12, 2007.
3 N. McKeown, T. Anderson, H. Balakrishnan, G.Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," ACM SIGCOMM Comput. Commun. Rev. (CCR), vol. 38, no. 2, pp. 69–74, 2008.
4. Open Network Foundation https://www.opennetworking.org/ in Proc. 4th ACM/IEEE Symp. ANCS, 2008, pp. 1–9.
5. B. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, future of programmable networks,"IEEE Commun. Surveys Tuts., vol. 16, no. 3, pp. 1617–1634, 2014.
6 . Mihai-Nicolae, Laura Gheorghe, Raluca-Andreea Somesan, " SDN-based Security Mechnism", IEEE, 2015, pp. 12-15.
7. Xiaolong Xu and Liuyun Hu, "A software defined security scheme based on SDN environment ", IEEE CyberC, 2017, pp. 504-511.
8. C. Yu, C. Lumezanu, Y. Zhang, V. Singh, G. Jiang, and H. V. Madhyastha. "Flowsense: Monitoring network utilization with zero measurement cost", In International Conference on Passive and Active Measurement - PAM. Springer, 2013.
9. R.J. Colvilleand G. Spafford, "Configuration Management for Virtual and Cloud Infrastructures", GartnerInc.,Oct.27, 2010 [Online] Available: http://www.gartner.com/id=1458131
10. A. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, and R. Sherwood, "On controller performance in software-defined networks," in Proc. USENIX Workshop Hot Topics in Management of Internet, Cloud, Enterprise Networks and Services (Hot-ICE), 2012.
11. E. Al-Shaer and S. Al-Haj, "FlowChecker: Configuration analysis and verification of federated OpenFlow infrastructures," in Proc. 3rd ACM Workshop Assurable and Usable Security Configuration, 2010, pp. 37–44.
12. M.Canini, D.Venzano, P.Peresini, D.Kostic, and J.Rexford, "A NICE way to test OpenFlow applications," in Proc. 9th USENIX Conf. Networked Systems Designand Implementation (NSDI), 2012.
13. A. Khurshid, W. Zhou, M.Caesar, and P.B. Godfrey, "VeriFlow: Verifying network-wide invariants in real time," in Proc. 1st ACM Workshop Hot Topics in Software Defined Networks, 2012, pp.49–54.
14. P. Porras, S. Shin, V. Yegneswaran, M. Fong, M. Tyson, and G. Gu, "A security enforcement kernel for OpenFlow networks," inProc.1st ACM Workshop Hot Topics in Software Defined Networks (HotSDN), 2012, pp. 121–126.
15. Purnima Murali Mohan et. al., "Towards Resilient In-band Control Path Routing with Malicious Switch Detection in SDN", IEEE COMSNETS,2018, PP.9-16.
16. S. A. Mehdi, J. Khalid, and S. A. Khayam, "Revisiting traffic anomaly detection using software defined networking," in Recent Advances in Intrusion Detection. Springer, 2011, pp. 161-180.
17. S. McGillicuddy, Microsoft Uses OpenFlow SDN for Network Monitoring and Analysis, TechTarget [Online]. Available: http://searchsdn.techtarget.com/news/2240181908/Microsoft-uses-OpenFlow-SDN-for-network-monitoring-and-analysis.
18. Dmytro Ageyev et. al., " Provision security in SDN NFV", IEEE TCSET. 2018, pp. 506–509.
19. R. Meyran, DefenseFlow: The First Ever SDN Application That Programs Networks for DoS/DDoS Security, Radware Blog, http://blog.radware.com/security/2013/ 04/defenseflow-dosddos-security/
20. S. McGillicuddy, Radware Adds Open Source DDoS Protection to OpenDaylight Project,TechTarget http://searchsdn.techtarget.com/news/2240203180/Radware-adds-open-source-DDoS-protection-to-OpenDaylight-Project
21. S. Shin, P. Porras, V. Yegneswaran, M. Fong, G. Gu, and M. Tyson, "FRESCO:Modular composable security services for software-defined networks," in Proc. ISOC Network and Distributed System Security Symp. (NDSS), 2013.
22. Wei Zhou, Li Li, Min Luo, Wu Chou, "REST API Design Patterns for SDN Northbound API", 28th International Conference on Advanced Information Networking and Applications Workshops, 2014.

AUTHORS PROFILE

Pradeep Kumar Sharma is presently working as Assistant Professor, in Computer Engineering at Manav Rachna International Institute of Research and Studies (MRIIRS), Faridabad. He is pursuing his Ph.D in Computer Engineering from MRIIRS faridabad. He did his M.Tech from MRIIRS and B.Tech in Computer Science and Engineering from UP Technical University.

Dr. S. S. Tyagi is presently working as a Professor, Computer Engineering and Dean at Manav Rachna International Institute of Research and Studies (MRIIRS), Faridabad. He completed his Ph.D in Computer Science and Engineering from Kurukshetra University, Kurukshetra. He did his M.E from BITS Pilani and B.Tech in Computer Technology from Nagpur University.

He is having an experience of more than 27 years in academics/teaching and research. He has been holding various academic and administrative positions during his career and contributing towards research. He is a senior member of various professional organizations like IEEE, ACM, CSI, QCI, ASQ etc. He is past chair, IEEE Computer Society, Delhi Section. There are more than 70 publications to his credit in National and International Journals. He is associated as an editor/reviewer of various journals. He has guided 05 Ph.Ds and several M.Tech Thesis and guiding Ph.D scholars in the field of Software Defined Networking, Cloud Computing, Adhoc Networks, Wireless Security etc.