International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 18 (2018) pp. 13475-13481
© Research India Publications. http://www.ripublication.com

Software Defined Networking: A Paradigm Shift in Networking for Future, Emerging Trends and Applications

Jayakumar Shanmugam

Indian Space Research Organisation, Sriharikota, India.

S.Thanga Ramya
RMD Engineering College, Chennai, India.

Abstract

Software defined Networking is an emerging topic in area of networking aimed at providing efficient and faster implementation of networking devices to support various applications in current world aimed at dynamic scaling of the network infrastructure to provide minimum time for upgrading as well as testing new protocols. The main aim of SDN lies in the separation of the currently co-existing control plane and data plane. This decentralisation involves providing greater leverage of access and intelligence to the control plane for controlling multiple network devices efficiently with minimum manpower. The control plane is designed such that the interactions with the application layer offer users to more programmatically control and define the network control. This papers aims at providing an insight into the concepts of SDN, the features of the SDN controller, applications of SDN and emerging areas of SDN implementation as well as points to be considered in developing efficient and secured SDN controllers.

INTRODUCTION

SDN concept has evolved out of the Clean Slate project from Stanford University to re-design the Internet architecture with a clean slate and to unplug the loop holes and inadequacies in the existing Internet domain. The project aimed at providing all the routing and access control decisions to a centralized controller and allowed the switching devices to simply carry out the tasks assigned by these controllers. This later paved the way for OpenFlow protocol developed by the Open Network Foundation (ONF). SDN is a standard which emerged from Open Networking Foundation (ONF) the using the OpenFlow protocol which focuses on providing more centralised control plane and decentralised forwarding data plane. It outlines the measures for the centralized network intelligence at the control plane. Hence OpenFlow protocol is the first protocol to shift the control out of the network switches and define the same using software algorithms which can suit to a variety of proprietary switches. By offloading the controllability, the switches were made to perform the packet forwarding faster at closer to wire speeds. SDN provides several advantages such as easier and efficient provisioning of network devices, ease of configuration using heterogeneous devices, control of the network from a centralized location, ease of integration with the cloud, provides network abstraction using the layered approach, lower capital and operating costs, ease of interfacing and operation with other technologies such as wireless networks and IoT devices, enhanced network security up to end point devices and better access to applications for controlling the entire network using software.

The paper is organized to present the SDN architecture with details about the various layers and the functions carried out in each layer. A section is devoted to Network Function Virtualization (NFV) and to define its closer association with SDN for implementing various solutions. Later topics revolve around the various applications of SDN such as the Big data analytics using Cloud computing, Network security, Network virtualization, Wireless sensor networks, IoT applications using SDN, and Green networking to list a few.

SDN ARCHITECTURE

The SDN architecture consists of the Application plane, Control plane and the Data / Infrastructure plane. The schematic of the SDN architecture is shown in Figure 1.

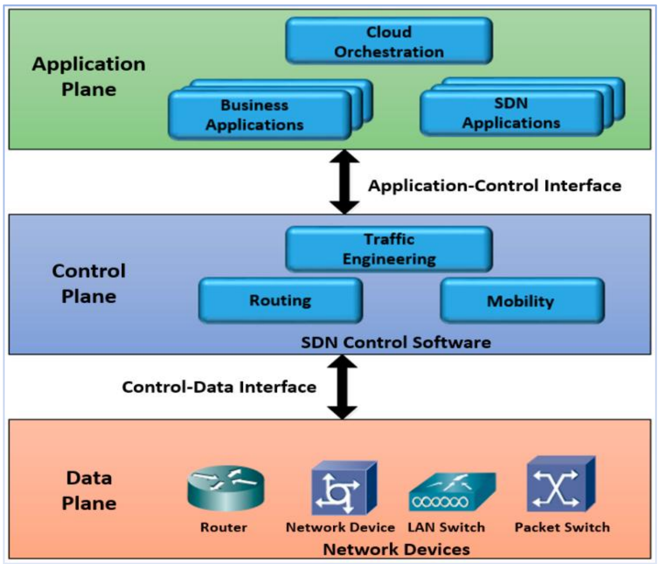

Figure 1: SDN Architecture

With the convergence of various technologies such as the Internet, Wireless sensor networks, Network Cloud and IoT, the requirement to interconnect to these devices and the smoother interaction by these devices with the user is ever increasing. For providing better services, common network architecture is required which can provide better access as well as controllability to such networks. SDN architecture provides the synergy for operating such networks having seamless interoperability.

SDN Control Plane:

SDN control plane houses the brain of the SDN, namely, the SDN controller which is primarily responsible for initiating, organizing and maintaining the network flow, along with performing other actions such as performing rule updates, network synchronization etc. This layer fits in between the SDN Application plane in the upward direction, named as North-bound interface and the SDN Data plane in the downward direction, name as South-bound interface. In the downward flow, it interacts with the SDN data plane and provides the packet forwarding rules and ensures consistency in the data flow. OpenFlow protocol provided by the CNF is the famous open source protocol for the south bound interface. In the upward flow, SDN controller interacts with the SDN application plane to provide vital information such as network synchronization status, network topology update, packet flow statistics and bandwidth occupancy.

The control plane creates the forwarding table entries which are used by the data plane for forwarding traffic between the ingress and egress ports of the network device. Initially a Routing information base (RIB) is created for storing the network topology which is consistent and loop free. A Forwarding information base (FIB) is created in the data plane for storing the forwarding table information in the switches. FIB is programmed once the RIB has attained consistent state. Once a packet is received by the control plane, the RIB gets updated and later updates the underlying FIB in the data plane to reflect the changes to all the network devices in the data plane. The routing decisions are done by the control plane whereas the forwarding plane performs only switching operations. As SDN can be deployed on large scale networks, there is a requirement for larger memory to store the packet forwarding rules along with proper cache replacement policies for efficient routing of packets with lesser packet drops. Memories such as Static Random Access Memory (SRAM) and Ternary Content Addressable Memory (TCAM) can be used for this purpose. SRAM offers easy scalability whereas TCAM performs faster access in packet classification. Efficient processing techniques can be implemented for packet forwarding which can segregate larger workflow to be implemented using specialised hardware whereas smaller workflows can be handled by the host CPU itself [1].

SDN controller design can be implemented using SDKs developed using high level languages such as C++, Java and Python. Cisco provides the One Platform Kit (onePK), whereas there are several other high level languages based on the network flow such as Frenetic [2], which provides SQL like declarative queries for obtaining the network traffic and statistics. The SDN applications generated provide network abstraction whereby the user is immune to the transactions happening at the lower level SDN data plane.

Network traffic status can be inferred by the SDN controllers using conventional way of probing till the last network device from the source node to the end node. This results in more overloading of the end node and hence reduced network performance at the lower nodes. Hence in SDN, optimization can be achieved by only probing larger network flows ignoring the smaller ones. Multiple SDN controllers can co-exist using the cloud whereby each SDN controller can be defined its territory of control. The SDN controllers can interact among themselves to provide a global view of the larger single SDN framework, wherein the network status of each SDN controller can be known to the others. Yin *et al.* proposed “SDNi” for interconnectivity and message exchange among multiple SDN domains [3]. SDNi provides a generic interface which coordinates the status and synchronization information. Examples of SDN controllers include OpenDaylight[4], NOX / POX[5], Ryu[6], Floodlight[7], Pyretic[8], Frenetic[2], Procera[9], Beacon[10] and Route Flow[11] for implementing software based network control.

SDN Data Plane:

The infrastructure layer consists of switches and routers controlled by the SDN controller using the south-bound interface. The data plane layer of the SDN is mainly devoted for packet forwarding. Packet forwarding in conventional networks is done using IP or MAC addresses, whereas SDN deploy other schemes such as forwarding based on TCP or UDP port, VLAN tags and ingress port switching. Traffic flow can be studied and based on the flow, large packet flows can be processed using ASICs for achieving faster packet forwarding where offloading the generic processor for processing smaller packet flows. The SDN switch implementation hence can be realized using a generic hardware such as PC which supports host OS such as Linux, or using a specialized hardware such as ASICs or using the vendor specific hardware. Enhanced topology discovery can be done using Tree Exploration Discovery protocol [13] to obtain the shortest path along with gathering topology information from the data plane without additional messages as compared to the LLDP protocol.

The existing limitation of configuring a maximum of 4096 VLAN IDs is overcome with the advent of VXLAN and an SDN overlay can be created which can service more than 4096 VLAN IDs as demanded in the case of cloud environment. The address space provided by the 24-bit segment ID of the VXLAN can create 16 million VLAN IDs. This provides more flexibility and elasticity to enables more number of users to get connected to the cloud and centrally controlled using the SDN controller. This feature provides greater impetus for geographically diversified users to migrate their applications to the cloud and possess a centralized control over the network infrastructure using the SDN technology.

An OpenFlow Logical Switch consists of one or more flow tables and a group table for performing packet inspection and forwarding. Also one or more OpenFlow channels exists which interface to the SDN controller. Using the OpenFlow switch protocol, the SDN controller can modify the flow entries in flow tables in real time. Each flow table consists of a set of flow entries, which contains matching fields, counters and a set of instructions. Matching starts from the first flow table and continues till the suitable match is found. Flow entries are forwarded on the ports which may be physical, logical or reserved ports. Reserved ports need not participate in the OpenFlow operations and hence carry out normal switch port operations based on the initial configuration. OpenFlow switches hence can function in two ways, namely OpenFlow only which support only OpenFlow operations, whereas OpenFlow hybrid switches support OpenFlow operations as well as normal switch operations. Logical ports perform operations such as link aggregation and tunnelling unlike normal switches whereas physical / virtual ports are defined by the switch which are mapped to the Ethernet interfaces. The OpenFlow switch contains at least one or more number of ingress flow tables, whereas egress processing is optional, based on the matching. If no egress table is configured as the first egress table, there is a packet mismatch and is considered as a table miss. After a table miss, options include dropping the packet, transmitting packet to another table or communicating back to the controller of the table miss. Consistency check is performed by the OpenFlow protocol for the tables used, flow control, group tables as well as the ports used for the operations [14].

SDN Application Plane:

SDN applications are interfaced to the SDN controller using the NorthBound interface. It consists of APIs such as virtual load balancer, virtual firewall, virtual traffic analyser, virtual intrusion detection systems and a host of many virtual network appliances to aid the SDN controller for smoother operations. Vendors also provide SDKs for users to experiment and evaluate using the APIs the SDN technology. There are a variety of SDN controllers in the market which use several APIs, which can be broadly classified into Representational state transfer (REST) APIs, adhoc APIs and some programming languages for building APIs. Examples of SDN controllers which use RESTful APIs include Floodlight, Instagram, Gmail, GitHub, OpenContrail and ONOS, whereas high level languages APIs are Frenetic[2], Nettle[12], Netcore[15], Pyretic[8], NetKAT[16] and Procera [9], to name a few.

SDN Management Plane:

There exists a management plane, apart from the three major planes in the SDN framework which is used for configuring the network devices initially and later is useful for monitoring the network traffic, assessing the traffic flows and hence acts an administrator to manage the network. It can be HTTP, SSH, SNMP. All interfaces, IP subnets, and routing protocols are configured either through CLI or northbound RESTful API. It also generates reports based on the data available and help network administrators to study the network and provide solutions in case of malicious behaviour.

NETWORK FUNCTION VIRTUALIZATION

As SDN can be considered being evolved by a team of researchers and data center architects, Network Function Virtualization (NFV) can be considered as being created by group of service providers for addressing the problems in configuring and positioning of network hardware. Network function virtualization is hence the phenomenon evolved for supporting big data centers and cloud environments with virtual network devices which are software configurable. With the invention of VmWare, the concept of IT virtualization of the server environment was possible and hence large data centers evolved where the server hardware supported various cross platform applications. Similar need for virtualization of the network hardware devices using software was required to support the pace of network expansion. Network function virtualization eliminates the use of installing and configuring proprietary network hardware by the use of software leading to virtual network products such as routers, switches, firewalls, network traffic analyzers and monitoring tools and many network appliances. This also helped in the network standardization using software which were earlier dominated by the proprietary hardware devices.

NFV is a technology developed for network standardization. This technology can exist independently without the use of SDN technology, wherein the network devices can be realized using software and integrated to virtual IT environment and provide easy access and control to the user. However, if both of them co-exist, then the SDN benefits such as separating the data plane from the control plane helps in tightly coupled centralized control of the control plane relinquishing the control over the data plane for packet forwarding by the devices. Hence the integration of SDN with NFV can be viewed as an added advantage for supporting smooth operations. Also by the use of open source protocols such as OpenFlow couples together to form the SDN-NFV-OpenFlow technology whose benefits can be reaped well by the NFV technology. This combination is going to be benchmark in the design of future networking technologies. This technology integration is as shown below:

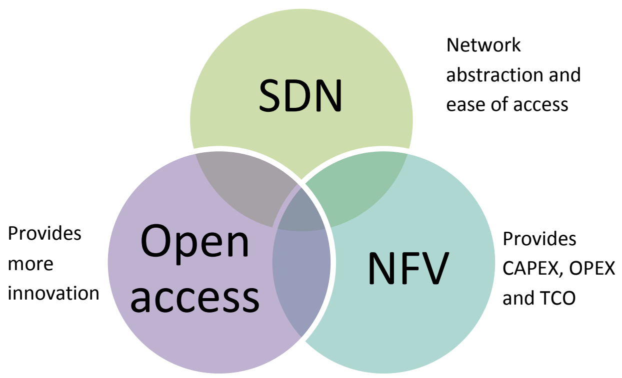

Figure 2: Inter-relationship between OpenAccess-NVF and SDN.

The realization of virtual network devices has led to the concept of white box networking helping in shaping the next generation IP infrastructure. European Telecommunications Standards Institute (ETSI), Open Networking Foundation, OpenDaylight Project[4], OPNFV and Linux Foundation are some organizations working on this next generation IP infrastructure. There are numerous applications using the NFV implementation by virtualization such as the VNIaaS, VNPaaS, VNFaas, mobile core network and base stations and CDNs.

Although the NFV is a very useful technology, there are some areas which still needs to be properly addressed such as Memory and CPU scalability, interoperability with heterogeneous networks, network security, scaling up to support IP end point devices, network reliability and management.

SDN APPLICATIONS

A. SDN for Cloud computing and Big data analytics

The extensive use of server and network virtualization over cloud environment has provided large scope for the use of SDN in the cloud computing. Big-Data Analytics can be deployed in the Cloud Data Centers(CDCs) to harness the massive computing power and extract information faster compared to the conventional methods. SDN can help Big-Data applications for better message passing among the cooperative nodes. With proper bandwidth allocation and priority assignments, sudden influx in the Big-Data flows can be effectively handled and thereby reducing the impacts on CDCs. The cloud infrastructure can be considered as IaaS which can be exploited to greater potential using SDN concepts. The data centres connected in the cloud can be geographically apart and when live VM migration is warranted across these data centres, it can be efficiently carried out using SDN solutions. Cooperative cache is an evolving technology in the Big data analysis and an SDN based cooperative cache network (SCCN) [17] for the IP networks is proposed by Yong Cui et al., which aims in minimizing the content transmission latency as well as reducing the inter-ISP traffic. A heuristic based algorithm performs intense computation for content placement decisions and later populates the new forward rule to SCCN Switches quickly for implementation. A cloud centric platform using SDN can be employed for analysing online social analytics [18] by understanding the perception of TV viewers for the various programs watched by them. A SDN system using Hadoop is used for performing quicker data transfer between the various processing units and the report the outcome to the SDN controller for enabling streaming of the contents desirous by most of the viewers.

With the advancements in the optical switching technology, SDN can be employed for providing solutions to distributed computing involving large data sets such as Big Data using the Hadoop framework and involves integrated network control, job scheduling, topology learning and configuring the optimal routes. This results in big performance improvement as illustrated by Guohui Wang et al., by programming the network at run time for performing Big Data applications [19] which employs traffic demand estimation of the Hadoop jobs and also employs network-aware job scheduling. SDN also plays an important role in the cloud services and content delivery networks. With the widespread use of cloud networks coupled with CDNs, it is becoming increasingly difficult to identify the traffic flows behind the various services. Martino Trevisan et al., used a concept called AWESoME [20], Automatic Web service manager, which uses big data algorithms for automatic learning and creating models describing the traffic of thousands of Web services as well build rules for steering the flows using SDN switches and hence provide a comprehensive way of managing web services.

B. Network Security using SDN

SDN solutions provide intrusion detection as part of secured SDN framework. Further security can be added by providing intrusion prevention systems based on open source SNORT [21] project. Also providing role based authentication provides access to select professionals to avoid disasters and have lesser down time due to human errors. Another advantage of SDN is that is has removed the apprehension of the placement of security appliances such as the perimeter firewall by routing all the information to the centralized SDN controller thereby providing secured data by extending the VLAN coverage area. Uses of data mining techniques provide efficient means for threat detection and mitigation [22]. Use of a RADIUS server can be considered along with the SDN controller for authentication of legitimate users. Jafarian et al. developed a Moving Target Defense (MTD) based SDN controller [23] where each host is provided with a virtual IP (real IP of the host remains to be statics) and is randomly mutated with higher rate for maintaining secrecy of the IP address. SDN controllers perform the IP translation between the real and virtual IP and preserve the network integrity.

Use of single SDN controller for controlling a big network creates additional overhead in terms of topology learning, updating port status, generating flow control and also providing network synchronization status to north bound interfaces. These additional overheads can have impact on the Quality of service (QoS) of the system. Also any intruder on gaining access to the centralized controller can implant Denial of Service (DoS) attacks whereby affecting the network services.

The management plane is designed to work outside of all the three planes defined by SDN. This plane is responsible for establishing network connection and perform network configuration. It is abstracted from any programming using the APIs for providing security and shielding the network from outside attacks. Although the North bound interface exists for providing authorization and authentication its role is still not clearly defined as the South bound OpenFlow. However, Porras et.al defined a role based authorization approach by proposing a security enforcement kernel [24] named FortNOX for the NOX OpenFlow controller which is robust enough to handle situation when the flow rules get changed for the OpenFlow application.

C. SDN for Wireless networks

Wireless networks is one area where already exist a centralized controller with distributed base station elements currently exist. Technologies such as Software Defined Radio (SDR) which uses the Generalized Multi-Protocol Labelling (GMPLS) can be integrated using the SDN technology. In a SDR, a large chunk of the wireless communication is designed and controlled using the software instead of realising using hardware, hence allowing the user to choose the nature of the radio signal to be generated and transmitted. Antenna designed using software provide high degree of directivity and reduce inter-signal interference.

OpenRadio[25] is a concept for designing programmable wireless network data planes which provides software abstraction to the lower physical and MAC layers. Hence with the advent of programmable data planes, digital transceivers could be developed with can be adaptive to various wireless protocol standards wherein a device can be configured to act as 3G or 4G or using Wi-fi service. In SDR, the physical layer activities are controlled using the software.

SDN controllers when integrated with the SDR's provide more efficient control over the bandwidth wherein the packet forwarding details can be more closely monitored to have minimum packet dropouts. User defined APIs can provide better programmability of various parameters such as bandwidth selection, QoS etc. Also SDN is going to play a major role in the shaping of the 5G technology which is to be user oriented instead of operator-oriented as in 3G or service-oriented as seen for 4G. As the 5G wireless technology revolves around ultra-high speed, low overhead, low delay access for cloud services, multi pathing is required for faster, reliable and secured data retrieval from cloud architectures. This multi-pathing is established using code centric networks [26], wherein the routers deployed perform compute and forward technique using random linear network coding. This provides efficient multi-path communication as well as dynamic allocation of distributed clouds on top of each router. SDN along with SDR and NFV provides a holistic solution for the 5G networks. SDN can provide solutions to overcome the existing limitations of multihop wireless networks for increasing the capacity of mobile systems. Networks such as Cellular SDN networks (CellSDN) [27] can maintain the Subscriber information base (SIB) for translating the subscriber rules to switching rules and allow easy reconfiguration of the network. CellSDN controllers may use Radio Resource Management(RRM) protocol [28] for northbound interfaces thereby providing simplified network operations. Use of distributed CellSDN controller can provide multiple parallel virtualized channels for communication wherein a failure of cellular architecture doesn't affect the entire network. Technology such as Fog computing are evolving in the processing of large data sets at the edge devices of mobile networks using SDN wherein the data centers are migrated from the core network to the edge network. The decoupling of the mobility management to the control plane and the processing to the data plane using Fog servers, efficient cell handover is carried out with less latency time and providing better QoS and QoE [29].

D. SDN and IoT applications

The conventional network devices such as switches, routers and other network devices do not have the programmability to interact with the new generation IoT devices. To overcome this limitation in conventional networks, SDN is the solution for such issues as it decouples the control plane from the data plane. The architecture of SDN consists of several application program interfaces (APIs), namely, the northbound and southbound interfaces. The northbound API interfaces the application layer with the control layer and provides the abstracted view of the network to the application layer. The southbound API interfaces the control and infrastructure layers and enforces different rules on the forwarding devices such as the routers and switches enabling rule based communication. OpenFlow is the most widely used protocol to enable communication between the control and data planes. The eastbound and westbound APIs also exist in the SDN framework. Eastbound APIs are used for connecting conventional IP networks with SDN networks. Westbound APIs are used for communication between the inter SDN domains. The Internet of Things (IoT) enables different objects such as sensor and RFID nodes, embedded systems and several intermediate devices to collect and exchange data using the SDN enabled switches. Various applications such as smart grids, planning of smart cities, smart health care and intelligent transportation systems use the IoT sensors for communicating with the master devices. Moreover, there exists several collaborative SDN-based technologies in IoT domain which are defined based on the type of networking service extended at various levels, namely at the edge, access, core and the cloud [30]. SDN-based IoT Mobile Edge Cloud Architecture (SIMECA) provides SDN support to IoT based mobile devices operating as Edge devices [31]. Since the future 5G technology is to widely provide wider access to IoT devices, the control of the IoT devices using SDN and NFV concepts assumes significance. In SIMCEA, the mobile edge devices are provided with distributed data planes to enable smoother IoT communication to the SDN controllers. In this method, the computing resources are positioned closer to the mobile edges to enable faster processing at the edge nodes. The technology proposes abstraction of the IoT service at both the control and data plane. At the data plane, the packet is made lighter using a smaller header and device identification s done only using the Device identity (DI) and the Routing identity (RI). When switching from one base station to other the Routing identity alone gets altered whereas the Device identity is left un-changed.

E. SDN and Green Networking

IEEE proposed the 802.3az standard which helps in producing Energy-efficient Ethernet (EEE) devices that allows physical layer transmitters to operate in low power mode during period of network inactivity. Green Networking is an evolving concept aimed at optimal use of the energy by various devices such as network switches, routers, servers and other components. Power consumption can be reduced in multiple ways such as designing CPUs which consume less power, powering off the networking devices as well use of reduced power for devices when the peak traffic is not in place. As SDN controllers are capable of analysing network traffic and providing efficient routing algorithms, they play a vital role in optimizing the power consumption of network devices in huge network infrastructure by alerting the idle devices to go into sleep mode or power-off mode. Examples of such areas are the huge data center and wireless mobile communications where the network traffic pattern keeps changing where the traffic during night time is very minimal. Also the traffic peaks only during certain period in a week or over a year during festive and holiday seasons. Hence, devices which are not in the critical path can be powered off without losing the redundancy, whereby reducing the power consumption contributing to green energy. Heller et al. propose an Elastic tree concept for green networking using SDN for determining minimum data links and switching devices required for a data center operation based on traffic conditions and provide dynamic load balancing [32]. Network traffic is controlled using an optimizer, power controller and a router. Various models such as Greedy bin-packerl, heuristic and prediction models can be used for creating an Elastic tree. Based on the link calculations, the number of switches required to be alive at the core layer and at the aggregation layer can be decided for supporting network traffic both upwards and downwards, leaving other switches to be either powered off or in sleep mode. SDN protocols such as OpenFlow can be used to determine the network traffic flow and hence provide a picture of the essential switches to be maintained in ON condition for the network flow to sustain without any disturbance. Other techniques involve designing energy efficient switches which work on low power mode during idle traffic conditions and also use of network proxy which keeps the network alive and allows the end hosts to go into sleep mode. In wireless mobile communications, based on the traffic load conditions, during lean periods, certain base stations can be powered OFF providing power optimization.

CONCLUSION

This paper provides an overview of the emerging field of Software Defined Networking which has already established it footprints in several applications. SDN is propelling the areas of Network virtualization, Cloud networking, next generation wireless networks, application using Internet of things, evolution and use of 5G user centric networks and various other fields to higher levels, by providing efficient services. SDN uses several open source protocols at its various planes and thereby removed the bottlenecks prevailing in using proprietary network hardware and firmware. The centralized operations provided by the SDN controllers provide more programmability to the data plane. The features such as mobility and WAN security drives more applications to use SDN solutions. There are certain issues which exist in the usage of the SDN technology such as proper definition of the NorthBound APIs, communication bottlenecks between the data plane and the SDN controller as well as protecting the SDN controller from various unknown threats, as its compromise will result in a single point failure and bring down the network operations to halt. However, these limitations will be overcome in the near future and pave the way of controlled and secured SDN operations.

REFERENCES

[1] G. Lu et al., “Using CPU as a traffic co-processing unit in commodity switches” Proceedings of 1st Workshop on HotSDN,2012, pp. 31–36.

[2] N. Foster *et al.*, “Frenetic: A high-level language for OpenFlow networks,” in *Proc. Workshop PRESTO*, 2010, pp. 6:1–6:6.

[3] H. Yin *et al.*, SDNi: A Message Exchange Protocol for Software Defined Networks (SDNS) across Multiple Domains, Jun. 2012, Internet draft. [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf

[4] Open Daylight Project URL https://www. opendaylight.org

[5] NOX / POX Controller URL: https://www.github.com/noxrepo

[6] Ryu Controller URL: osrg.github.io; Ryu SDN Framework Community 2014.

[7] Floodlight Project URL www.projectfloodlight.org; Big Switch network sponsored community project

[8] J. Reich et al., “Modular SDN Programming with Pyretic,” USENIX: vol. 38, no. 5, October 2013.

[9] A. Voellmy, et al., “Procera: a language for highlevel reactive network control,” in Proceedings of the first workshop on Hot topics in software defined networks, ser. HotSDN ’12. New York, NY, USA: ACM, 2012, pp. 43–48.

[10] The Beacon OpenFlow Controller; Proceedings of 2nd ACM SIGCOMM workshop on Hot topics in Software Defined Networking, Aug 2013.

[11] Route Flow Controller Project URL: https://www.github.com/routeflow/RotueFlow; https://www.sdxcentral.com/listings/routeflow-project

[12] A. Voellmy et al., “Nettle: taking the sting out of programming network routers,” in Proceedings of the 13th international conference on Practical aspects of declarative languages, ser. PADL’11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 235–249.

[13] Elisa Rojas et al., TEDP: An enhanced topology discovery service for Software-Defined Networking, IEEE Communication Letters, Vol 22, Issue 8, Aug 2018.

[14] OpenFlow Switch Specification version 1.5.1, Open Networking Foundation, March 26, 2015.

[15] C. Monsanto et al., “A compiler and run-time system for network programming languages,” SIGPLAN Not., vol. 47, no. 1, pp. 217–230, Jan. 2012.

[16] C. J. Anderson et al., “NetKAT: Semantic foundations for networks,” SIGPLAN Not., vol. 49, no. 1, pp. 113–126, Jan. 2014.

[17] Yong Cui et al., SDN-based Big Data Caching in ISP Networks. Published in IEEE Transaction on Big Data (Early Access), Jan 2017.

[18] Han Hu et al., Toward an SDN-Enabled Big Data Platform for Social TV Analytics. Published in: IEEE Network (Volume: 29, Issue: 5, September-October 2015).

[19] Guohui Wang et al., Programming Your Network at Run-time for Big Data Applications. HotSDN’12, August 13, 2012, Helsinki, Finland.

[20] Martino Trevisan et al., AWESoME: Big Data for Automatic Web Service Management in SDN. IEEE Transactions on Network and Service Management, VOL. 15, NO. 1, MARCH 2018.

[21] S. project. URL http://www.snort.org/

[22]. K. Cabaj, K. Szczypiorski, S. Becker, “Towards Self-defending Mechanisms Using Data Mining in the EFIPSANS Framework,” Advances in Multimedia and Network Information System Technologies, Advances in Intelligent and Soft Computing, nr 80, 2010, Springer, pp. 143-151.

[23] J. H. Jafarian, et al., “OpenFlow random host mutation: Transparent moving target defense using software defined networking,” Proceedings of 1st Workshop on HotSDN, 2012, pp. 127–132.

[24] Porras, Philip, et al. "A security enforcement kernel for OpenFlow networks," Proceedings of the first workshop on “Hot topics in software defined networks,” ACM, 2012, http://dx.doi.org/ 10.1145/2342441.2342466.

[25] Manu Bansal et.al, OpenRadio: A Programmable Wireless Dataplane, Stanford University. Proceedings of SIGCOMM 2012 on “Hot topics in software defined networks,” pp.109-114.

[26] ND Szabo, F Nemeth, B Sonkoly, A Gulyas, FHP Fitzek, ” Towards the 5G Revolution: A Software Defined Network Architecture Exploiting Network Coding as a Service”, Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, August 2015.

[27] L. L. Erran, M. Z. Morley, J. Rexford, Cellsdn: Softwaredefined cellular networks, Tech. rep. (2012).

[28] J. Gozalvez, M. C. Lucas-Esta˜n, J. Sanchez-Soriano, Joint radio resource management for heterogeneous wireless systems, Wirel. Netw. 18 (4) (2012) 443–455.

[29] Yuanguo Bi et.al, Mobility Support for Fog Computing: An SDN Approach, IEEE Communications Magazine May 2018, pp.53-59.

[30] Samaresh Bera et al., Software-Defined Networking for Internet of Things: IEEE INTERNET OF THINGS JOURNAL, VOL. 4, NO. 6, DECEMBER 2017

[31] Binh Nguyen et al., SIMECA: SDN-based IoT Mobile Edge Cloud Architecture Published in 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)

[32] B. Heller et al., “ElasticTree: Saving energy in data center networks,” in Proc. 7th USENIX Conf. NSDI, 2010, p. 17.