Towards OpenFlow Based Software Defined Networks

Pallavi Chhikara
Department of Information Technology
Amity University Uttar Pradesh
Noida, India
chhikara.pallavi@gmail.com

Gurpreet Singh Matharu
Department of Information Technology
Amity University Uttar Pradesh
Noida, India
mtech.gurpreet@gmail.com

Vikas Deep
Department of Information Technology
Amity University Uttar Pradesh
Noida, India
vdeep.amity@gmail.com

Abstract - Software defined networks (SDN) are an emerging technology that is being increasingly adopted by various network operators. These technologies provide new services and powerful analytics that help to transform the network and unfasten its intelligence to serve today's business demands. This paper briefs about the need for change in the current networking technology and explores the role of Open Flow protocol that is used by researchers to experiment with more realistic settings to provide for a new network architecture. Further, this paper discusses the advantages offered by SDN and the huge potential of OpenFlow based SDN. As SDN can simplify management of virtualized networks, enable cloud computing and reduce costs, the vendors would be encouraged to adopt SDN and OpenFlow. The objective of this paper is to provide an insight into the latest technology to the vendors to assist them in future enhancement of their switch products in the network.

Keywords - Software defined networks (SDN); OpenFlow; SDN Architecture; OpenFlow Working; OpenFlow Switch

I. INTRODUCTION

In the present technological scenario, computer networks are being built by implementing complex protocols on an array of network devices such as routers, switches and middle boxes like intrusion detection systems and firewalls. In order to allow the network to adept to the changing environment, the network operator manually configures all new configuration policies by converting them into the low level commands that are supported by the system. Hence, network buyers need to depend on the network vendors in order to deal with the problems related to security, mobility, manageability, virtualization and other issues. Many efforts have been done to make the network programmable even before the advent of SDN. Some of those are mentioned in [13] such as 4D project, NETCONF, Ethane.

Software Defined Networks (SDN) is a new programmable networking architecture that separates the network control plane from the data forwarding plane and hence simplifies the network management by providing applications and services with the abstracted centralized view of the network topology and its states. SDN provides a software abstraction layer on top of physical infrastructure, thereby hiding the complexity and providing transport layer visibility to applications and services. Through its increased intelligence and open environment, it provides virtualization of the network and centralization of network operation and also helps in rapid development of applications. A logically centralized controller in the control plane is responsible for taking all the control decisions using the global network state and it communicates with the networking elements via standardized interfaces. The most commonly used protocol in SDN is OpenFlow procotol, which allows communication between the data plane and control plane by providing an interface between them [13].

SDN, instead of directly configuring each of the connected devices in the network, allows the network administrator to dynamically establish multiple networks. Using high level control programs, it can allocate bandwidth and route data for optimized performance [2]. SDN allows flow control at the network level by abstracting it from individual devices. This data flow control at network layer allows the administrator to define network flows, based on the needs of different groups of users and connectivity requirements of end stations [1]. It consists of three primary areas:

1) *Network virtualization layer (data plane):* This layer will physically carry data packets from one network device to another depending on the rules being applied in the device hardware.
2) *Value added services layer (control plane):* This layer provides programmatic control APIs and contains the logic for the devices to program the data plane and provide control of specific traffic flows to the network operator.
3) *Cloud management layer (management plane):* The orchestration layer helps network operator to use standards based interfaces and plug ins such as OpenStack, CloudStack and vCentre to control the policies of their networks [15] [4].

Two commonly known systems implementing the SDN principle are OpenFlow and ForCES (Forwarding and Control Element Separation) [13]. ForCES protocol works similar to the master slave model in which forwarding elements (FEs) act as slaves and they are controlled by control elements (CEs) which act as masters. The main functional block of ForCES protocol is logical Function block (LFB).Another well defined protocol used in SDN is OpenFlow that allows the administrator to select the path for data flow in a network. At present, SDN is being implemented through OpenFlow by various global vendors including IBM [1]. Comparison between ForCES and OpenFlow is given in [14].

The rest of this paper has been structured as follows: Section II focuses on limitations of current networking technology followed by Section III which emphasises on the need for a new network architecture. Section IV elaborates on the architecture of SDN followed by Section V which provides an overview on OpenFlow and Section VI discusses the OpenFlow switch components along with its working. Further, Section VII mentions the advantages offered by SDN followed by Section VIII which briefs upon the advantages offered by OpenFlow based SDN. Section IX deals with OpenFlow vulnerability followed by Section X which outlines the limitations of SDN, ending up with Section XI which summarizes the paper.

II. LIMITATIONS OF CURRENT NETWORKING TECHNOLOGY

Some of the limitations being posed to the current networking technology are as follows [7]:

- *Complexity*: Present day networks are relatively static as the protocols are usually defined independently for solving a specific problem and the networks cannot be modified dynamically as per the changes in traffic, user demand and applications.
- *Contradictory policies*: Due to complexities in the current networks, it becomes difficult to apply a same set of QoS, access control, security parameters and other policies to an increasing number of mobile users and this leads to security breaches and other negative consequences.
- *Lack of scalability*: As the need for data centers is growing day by day, it demands for the network to grow as well. Achieving such scalability in order to provide the users with high performance and low cost connectivity to the servers is not possible with the manual configuration.
- *Multi tenancy*: The implementation of multi tenancy comes out to be very complex in the existing environment as the network needs to serve multiple users with varying applications and varying performance needs that demands for specialized devices, further adding up to capital and operational costs.
- *Vendor reliance*: Lack of standard open interface restricts the capability of network operator or researcher to change the network according to their individual requirements and environment.
- *Underutilized resources*: Some of the network resources remain underutilized in the resource pool that leads to huge waste of investments. This problem can be solved by network virtualization that allows resources to be virtually (logically) places in a resources pool and can be accessed whenever needed.
- *Management cost:* Whenever a new device is added or removed, it has to be configured individually or on switch by switch basis that will add up to management and labor costs.

III. NEED FOR A NEW NETWORK ARCHITECTURE

Conventional networks were hierarchical in design and the switches were arranged in a tree structure which was suitable for client server computing, but the present competitive environment, with dynamic computing and storage, demands for a more flexible architecture. Following are the few reasons which motivate us to shift to a more programmable architecture:

- *Changing traffic*: Of late, communication has not just remained restricted to client and single server rather the applications are now accessing a plethora of databases and multiple servers. The users affect network traffic by accessing content and applications dynamically from anywhere, at any time. In addition to that, enterprises are offering computing models including private clouds, public clouds or a combination of both, resulting in additional traffic in WAN [4].
- *Different switches*: They differ internally resulting in incompatibility. Moreover, vendors do not like to open interfaces within their boxes as they fear that new experiments may bring in unintended competition for their protocols and crash down the networks.
- *Increase in cloud services*: Enterprise businesses want to access applications and resources on demand. Whether in public or private cloud, providing self services requires scaling in storage, computing and network resources.
- *Big Data requires more bandwidth*: Constant rise in data demands for more network storage in data centers and it also requires better parallel processing amongall the directly connected servers [4].

IV. ARCHITECTURE OF SDN

In SDN architecture, network control is decoupled from the data plane and software and applications running in the control plane can programmatically manage the network resources. The network's intelligence is placed in a centralized software based controller, which has the global knowledge of the network and makes the network appear as a single, logical switch to the application. SDN allows the network devices to directly accept the instructions from the controller rather than having to understand and process different protocol standards.

By providing centralization in the control layer, it allows network managers to flexibly configure, secure, manage and optimize resources with the help of dynamic, automatic SDN programs. SDN architecture support APIs ensuring common network services like quality of services, routing, access control etc. SDN architecture is built on three layers [9]:

1) *Infrastructure layer:* It is the foundation layer that comprises of both physical and virtual network devices such as virtual switches, routers and switches. All the network elements use OpenFlow protocol to implement traffic forwarding rules.

2) *Control layer:* It consists of a centralized controller that abstracts the physical infrastructure to provide a global view to the entire network. This layer uses OpenFlow protocol to communicate with the upper layer i.e. infrastructure layer.

3) *Application layer*: This layer compromises of network services, application and orchestration tools that are used to interact with the control layer. It provides an open interface to communicate with other layers in the architecture.

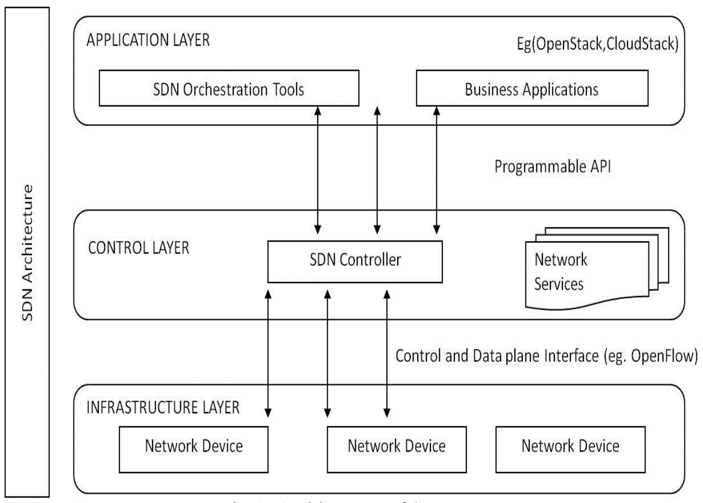

Fig.1. Architecture of SDN

V. OPENFLOW OVERVIEW

OpenFlow was developed in 2007 by Stanford University and University of California at Berkeley. It was defined as a standard by Open Networking Foundation (ONF) in 2011. The specification for OpenFlow is controlled and published by Open Network Foundation (ONF), which has more than 60 major companies as its members including Facebook, Google, Microsoft etc. [8].

OpenFlow is a standard interface that is employed to communicate between control plane and forwarding plane of SDN and also allows for programming the data plane switches. OpenFlow protocol provides centralized control of the networking devices like switches, routers etc. in a virtualized manner and hides the complexity of individual network devices, thus simplifying network management. It allows for easy understanding of the whole network and optimization of resource utilization by making them available to meet the varying workload requirements within the network. As OpenFlow is programmable, it allows the network manager to configure different policy rules corresponding to each set of devices and users [6]. Open Flow based controllers discover and maintain an inventory of all the links in the network and then create and store all possible paths in the entire network [8]. Unlike the OpenFlow that allows the network to react to real time changes, in current IP based routing all the flows follow the same path between two endpoints without taking into account the different requirements of different users.

OpenFlow protocol overcomes the drawback of current IP networks, which lack real-time configuration, and provides a broader view of the overall network. OpenFlow protocol can instruct switches and routers to direct the traffic by using software based controllers to manage the flow tables to quickly change the network layout and traffic flows. Controller makes all the forwarding decisions using the complete network view, monitoring of resources and network virtualization. This allows OpenFlow protocol to provide QoS, such as performance by choosing the required bandwidth, scalability by configuring all the new network devices at once using the controller and reliability by dynamically changing the flow. End-to-end QoS for multimedia delivery has been described in [5].

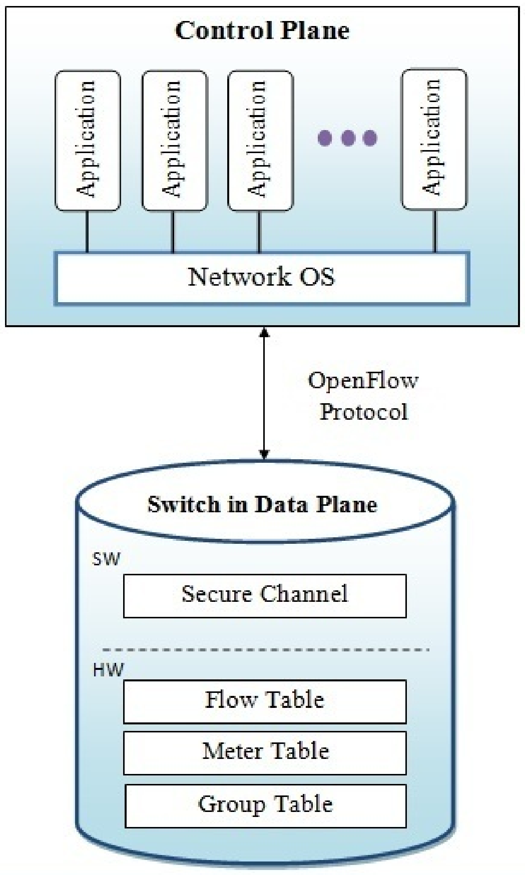

Fig.2. OpenFlow Switch

Controllers can manage different set of switches as a single virtual switch. Controllers and applications together act as network's operating system, which can be used by the network operator to implement SDN. Control plane has a network operating system which manages the application running on it. These applications provide the major network services like routing, authentication, security, quality of services etc. The network operating system (OS) communicates and receives information from the data planes using the secure channel and OpenFlow protocol and then applies the actions as requested by the applications [10].

TABLE 1. VARIOUS VERSIONS OF OPENFLOW RELEASES WITH ADDED FEATURES

Version	Features	Year
1.0	Slicing by allocating bandwidth to queues, flow cookies, user specifiable data path description	2009
1.1	Support for MPLS, multiple tables, multiple path, virtual ports, VLANs,	2011
1.2	Support for extensible headers and matches, IPv6, multi layer switching, flexible flow counters, OpenFlow over SCTP	Dec 2011
1.3	Support for tunneling, per flow traffic meters, provider backbone bridging tagging	2012

VI. OPENFLOW SWITCH COMPONENTS AND ITS WORKING

OpenFlow network consists of two types of switches, one of which is a pure switch that supports only OpenFlow operations and relies on the controller for all the forwarding decisions, whereas the other one is a hybrid switch that supports the normal Ethernet switching operation in addition to OpenFlow operations.

An OpenFlow switch consists of 3 main components (1) flow tables those are used to perform the matching and forwarding functions, (2) OpenFlow channel that provides the interface to connect each switch to the controller so that commands and packets can be sent among them and it is usually encrypted using TLS for proving security, (3) OpenFlow protocol that is used for communication between switch and controller by passing three types of messages i.e. controller to switch messages which may or may not require responses from the switch, asynchronous message which are sent from the switch to the controller upon change in the states like new packet arrival, and symmetric messages which can be send in either direction without solicitation [16].

Each switch consists of atleast one flow table or more and each flow table contains flow entries, which specifies how the packets would be processed and forwarded. Controller will use the global view of the network to add or update or remove a flow entry from the flow table and flow entry can also be removed upon expiry. Each of the flow entries consist of:

1) Headers or match fields that are used to match the incoming packets for source address, destination address, ingress port and metadata (used to pass information between the tables) fields.

2) Set of instructions or actions that is applied on finding a match in the flow entries and will determine who the packets have to be processed.

3) Counters that store the statistical information about the particular flow such as duration of the flow, number of received packets and number of bytes received [1].

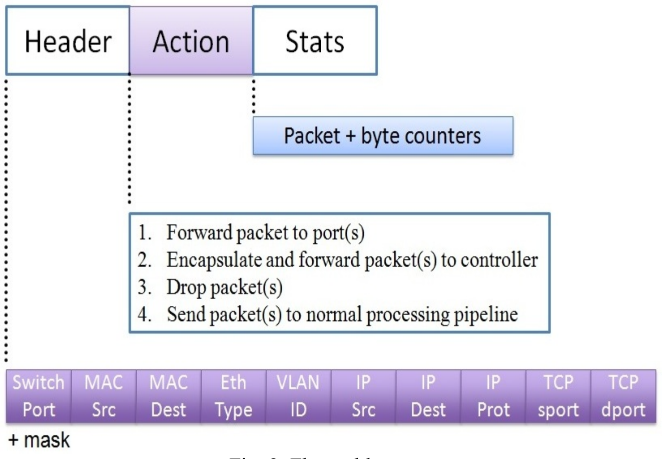

Fig. 3. Flow table entry

Packets are matched against multiple flow tables in a pipeline where the tables are sequentially numbered beginning from 0. Other flow tables are looked up depending upon the output from the previous table. After the appropriate match is found, the action in that entry may be applied or action can be a Goto instruction, which will forward the packet to the next table. If the packet does not match with any of the flow entries, then the action depends on the table miss entry in the flow table. A table miss entry is present in every flow table, indicating the actions that can be applied to the unmatched packets. Action can include dropping the packet, passing it to another table or sending the packet to controller using the secure channel. Table miss flow entry is identified by making all the match fields as wildcard entries and it has the lowest priority number.

VII. ADVANTAGES OF SDN

Advantages offered by Software Defined Networks (SDN) include:

- As SDN can control data flows, it would route traffic more efficiently in networks using the path having fewer hops and more available bandwidth. It can help in creating a secure and private path in the network for some specific data flows [1].
- SDN would allow the administrator to maintain virtual networks over the physical networks containing specified compute and storage resources. Also, these virtual networks would make it easy to handle virtual workloads among different data centers to provide optimized, scalable network as demanded by the business [3].
- Load balancing is supported by SDN between OpenFlow switch and server and this would improve the performance of the network.
- To meet the dynamically changing application and service requirements, SDN networks allow for automatically expanding the network bandwidth.
- SDN would also reduce the management cost (OPEX and CAPEX) by using OpenFlow to configure the network elements automatically rather than network operators performing the configuration tasks manually.
- Network slicing is possible by using the controller to create subnetworks that can serve specified workloads in order to increase the performance and make optimized used of network resources. For example subnetworks can be configured to avoid the variation and jitter in order to serve the video workloads efficiently.
- SDN provides layered architecture with standard open interfaces, which allows independent innovation at each layer, thus providing the scope for conducting experiment and research more efficiently by utilizing dedicated bandwidth.
- SDN provides flexibility by allowing easy customization and integration with other software applications and also by supporting fast upgradation by programming the network using software.

VIII. ADVANTAGES OF OPENFLOW BASED SDN

Benefits that the business enterprises and customers can achieve by deploying SDN architecture include:

- *Centralized control*: SDN controlleris capableof managing any network device that is having OpenFlow, independent of its vendor. SDN based tools can be used to configure and update the entire network at the same time.
- *Provides automation to reduce complexities*: SDN provides a flexible automation and management

framework though which tools can be developed that can perform the management task automatically rather than performing themmanually. These tools can support emerging IT as a service models and reduce the operational overhead and instability caused by operator errors.

- *Platform for innovation*: SDN allow researchers to experiment their innovations in real time in virtualized slices of production network based on their own requirements rather than relying on laboratory test beds that are small for denoting real world networks. This allows them to adjust the performance and bandwidth required from the network and also initiates their new innovative services and capabilities in a short span of few hours instead of days and months.
- *Provides reliability and security*: It reduces the possibility of failures in network due to policy or configuration inconsistencies as it allows addition or removal of applications, endpoints and policies without the need to configure them individually.
- *Better network control*: OpenFlow based SDN provides abstraction and automation and allows IT to provide policies at granular levels such as session, devices, user and application levels. This type of control allows the network operator to support multi tenancy by separating the traffic of individual customers based on their data requirements and managing the resources among multiple customers sharing the samenetwork.
- *Better experience*: By providing centralization and abstraction, SDN architecture can adapt better to dynamic user needs. For example, if a user wants to use a video service with the highest possible resolution available, then unlike the present day networks that introduce delays and interruptions in users' experience, OpenFlow based SDN allows the video based applications to identify the real time bandwidth availability offered by the network and then accordingly adjust the resolution of the video.

IX. OPENFLOW VULNERABILITY

There are numerous vulnerabilities in the Open Flow network designs due to separation and centralization of the control plane. Open Flow is vulnerable to man in the middle attacks due to the failure to adopt TLS (transport layer security) for Open Flow control channel in switches and controllers by the vendors. Open vSwitch is the only controller with full TLS support [2].

Open Flow allows for customization of the network by separating the data flow from the control plane. The controller configures the switches in data planes to forward the packets by installing rules implementing the Open Flow protocol. The controller can install rules either when it receives the traffic i.e. reactively or the rules can be installed immediately to handle the traffic i.e. proactively. Controllers and switches communicate over a TCP connection that can or cannot be protected by TLS with mutually authenticated certificates that are signed by a private key corresponding to a mutually agreed public key.

The switch firstly establishes the TCP connection with operator configured settings that contain the IP address and TCP port of controller. FlowVisor is present in between the controller and switch to apply limitations to all those rules created by the controller. Some of the vulnerabilities associated with OpenFlow are discussed [2]:

- *Man in the middle attack:* In normal networks, in order to capturethe credentials, an attacker has to wait for the network operator to log into each switch management interface whereas in the OpenFlow networks, due to constant connectivity and lack of authentications in plaintext OpenFlow control channel, an attacker can immediately seize full control of switches and execute the eavesdropping attacks that are difficult to detect.
- *Switch authentication*: Even after TLS has been implemented, failure to implement switch authentication in the controller can allow the attackers to gain information about the network by understanding the controller's discovery logic and observing how the controller responds to different packets and then the attackers can take down certain portions of the network or gain sensitive information from it.
- *Flow table verification:* Even though implementation of TLS increases the security but it does not help in detecting the switches which erroneously alter the rules and this mismatch between the rules generated by the network and actual rule state may lead to access control failure. The only way to verify these rules is to inspect the flow tables from each switch; although this method comes out to bevery costly computationally.

X. LIMITATIONS OF SDN

SDN also brings along several new security issues which must not be ignored. Further, researches are being conductedto study the impact of these security issues on the networks. Also, attempts are being made to overcome these issues. Some of those security issues have been discussed:

- *Denial of service (DOS) attack:* In SDN, data plane can demand for new flow rules of unmatched packets from the control plane. This may cause serious problems as the data plane receiving too many unmatched requests in a short time interval would forward all those requests and flood the control plane, hence causing DOS attacks. Similarly, fake requests would force the control plane to generate new flow rules for the data plane and hence making it difficult for the data planes to store rules for the normal flows [11].
- *Single point of failure:* Centralized controller acts as a single point for handling the data flow in data plane leading to several issuessuch as scalability and reliability and therefore a distributed controller has been proposed making the SDN architecture logically centralized but using physically distributed controller. One such solution has been proposed in [12].
- *Compatibility with legacy architecture:* Another issue is how to migrate to SDN architecture while still protecting

the cost of investment in the existing networking equipment.

- *Detecting the comprised controller:* As all the control is placed within the controller, it becomes very important to secure the controller to prevent unauthorized access and attacks on the network, hence techniques must be in place to detect the attacks on the compromised controller.

XI. CONCLUSION

In the rapidly evolving networking scenario, networking would tend to rely greater upon software to boost up innovation in the field of networks. This paper emphasizes on the requirement for innovative network architecture and explores all potential of OpenFlow based software defined networks. Further, this paper encourages the adoption of OpenFlow and SDN among vendors. SDN is capable of transforming present day static networks into more flexible, dynamic platforms that can intelligently and dynamically program the allocation of resources and provide scalability required to support large data centers. It can also provide virtualization that is required to support automated, dynamic and secure cloud environment. Further, it can provide a dynamic network that transforms the traditional networks into those with a rich set of service delivery platforms. OpenFlow is one approach for implementing SDN and various others approaches can also be used. As the SDN is going through an evolving phase, it is advisable to process it through prototype effort and evaluate it before its adoption. Although SDN is in its infancy, it is expected to make our network more flexible, dynamic and bring a reduction in the operational complexity.

REFERENCES

[1] N. McKeown, T. Anderson and H. Balakrishnan, "OpenFlow: Enabling Innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69-74, April 2008. DOI = 10.1145/1355734.1355746

[2] K. Benton and C. Small, "OpenFlow vulnerability Assessment," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking (HotSDN'13)*, Hong Kong, China, 2013, pp.151-152. DOI = 10.1145/2491185.2491222

[3] IBM Systems and Technology, "IBM software defined network for virtual environments," *White Paper*, June 2013. Internet: http://www.research.ibm.com/haifa/dept/stt/papers/QCW03028USEN.PDF

[4] Brocade Communications Systems, Inc., "Network Transformation with Software Defined Networking and Ethernet Fabrics," 2012. Internet: http://community.brocade.com/dtscp75322/attachments/dtscp75322/SoftwareDefined/49/1/network-transformation-sdn-wp.pdf

[5] H.E. Egilmez, S.T. Dane, K.T. Bagci and A.M. Tekalp, "OpenQoS: An OpenFlow Controller Design for Multimedia Delivery with End-to-End Quality of Service over Software-Defined Networks," in *Asia-Pacific Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Hollywood, USA, 2012, pp. 1-8.

[6] Extreme Networks, Inc., "OpenFlow and SDNs," *White Paper*, 2012. Internet: http://learn.extremenetworks.com/rs/extreme/images/Open-Flow-and-SDNs-wp.pdf

[7] Open Networking Foundation, "Software Defined Networking: The New Norm for Networks, *White Paper*, April 2012. Internet: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[8] HP, "Prepare for software defined networking," *White Paper*, 2013. Internet: http://h20195.www2.hp.com/v2/GetPDF.aspx/4AA3-8562ENW.pdf

[9] Open Networking Foundation. Internet: https://www.opennetworking.org

[10] P. Fonseca and R. Bennesby, "A Replication Component for Resilient OpenFlow based Networking" in *IEEE Network Operations and Management Symposium (NOMS)*, Maui, USA, 2012, pp. 933-939. DOI = 10.1109/NOMS.2012.6212011

[11] S. Shin and G. Gu, "Attacking Software Defined Networks: A First Feasibility Study" in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking (HotSDN '13),* Hong Kong, China, 2013, pp. 165-166. DOI = 10.1145/2491185.2491220

[12] A. Dixit and F. Hao, "Towards an Elastic Distributed SDN Controller" in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking (HotSDN '13),* Hong Kong, China, 2013, pp. 7-12. DOI = 10.1145/2491185.2491193

[13] B.A.A. Nunes and M. Mendonca, "A Survey of Software Defined Networking: Past, Present, and Future of Programmable Networks," IEEE Communications Surveys & Tutorials, vol. 16, no. 3, pp. 1617-1634, Feb. 2014. DOI = 10.1109/SURV.2014.012214.00180

[14] Z. Wang, T. Tsou and J. Huang, "Analysis of Comparisons between OpenFlow and ForCES," *Internet Draft*, 2012. Internet: https://tools.ietf.org/html/draft-wang-forces-compare-openflow-forces-01

[15] Brocade Communications Systems, Inc., "Network Transformation with Software-Defined networking and Ethernet Fabrics," 2012. Internet: http://community.brocade.com/dtscp75322/attachments/dtscp75322/SoftwareDefined/49/1/network-transformation-sdn-wp.pdf

[16] Open Networking Foundation, "OpenFlow Switch Specification", 2012. Internet: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf