I J C T A, 9(37) 2016, pp. 151-157

Enhancement of OpenFlow to Improve the Performance of SDN

Rakshith K Salian[*] and C T Manimegalai[**]

Abstract: There is a potential increase in growth of the communication network in terms of the size of the network as well as the complexity in the traditional network system infrastructure and the protocol stack cannot fulfil all the contemporary networking demands by providing adequate solution. Hence easier methodology has to be introduced so that new abstractions can be created in an affable manner resulting in ease of network management. The key trend in the cloud architecture is the core infrastructure that facilitates the storage, networking and the computing resources required is becoming software defined. As far as the virtualization and the network resource management is concerned SDN plays a prominent role in an on demand manner. SDN is the main backbone of IoT which offers centralizing controlling, abstracting the network devices, and providing flexibility, dynamic and automated reconfiguration of the network. There are many research challenges the remain on how to furnish the demanded quality of service, optimal load balancing, extent the scalability, throughput and deliver the stipulated security for the network. Hence this research generally realize the delay that is caused by the SDN controller and provide the optimal solution for this issue. Hence this research focus on improving the processing speed and thereby the throughput of the OpenFlow by having an optimal solution for the SDN controller, hence enhancing performance penalties.

Keywords: Cloud, IoT, OpenFlow, Software Defined Network (SDN), Virtualization.

1. INTRODUCTION

Nowadays Software which enables the Network to be Programmable and virtualizable which in turn makes the network operation to accomplish the task in a simple and reliable manner. In the old traditional technologies, the equipment such as switches and routers were limited only to reading address and transferring packets to the other system. But the Telecommunications networks are undergoing major and tremendous improvement so as to meet the requirements of the next generation of services and the Users, which results in a design which helps in need for a general revised architectural approach rather than a series of local and incremental technology updates[1]. SDN plays an important role in high traffic applications such as the video transmission and cloud based application in the mobile broadband networks.

The Open Networking Foundation (ONF) is one of the nonprofit consortium which is mainly dedicated to the standardization, development, and commercialization of SDN [2]. ONF has contributed the most explicit and well acceptable definition of SDN given as follows: Software-Defined Networking (SDN) is an emerging network architecture where network control is decoupled from forwarding and is directly programmable [3]. As per this definition given by ONF, SDN is characterized by two entities, namely decoupling of control and data planes, and programmability on the control plane. Nevertheless, neither of these two signatures of SDN is totally new in network architecture.

The main attribute of the SDN is its inheritance of decoupling the data plane and the control plane from the underlying architecture which in turn results in offering a greater control of a network through programming technologies. The combination of these features will bring out the potential benefits of enhanced configuration, improved performance, and encouraged innovation in network architecture and operations. The control incorporated by the SDN may include not only packet forwarding at a switching level but also the link tuning at a data link level, by penetrating the barrier of layering. SDN also has the

---

[*] PG Scholar, Department of Telecommunication Engineering, SRM University, Chennai. *Email: rakshith_salian@srmuniv.edu.in*

[**] Professor, Department of Telecommunication Engineering, SRM University, Chennai. *Email: manimegalai.c@ktr.srmuniv.ac.in*

ability to acquire instantaneous network status, SDN permits a real-time centralized controlling of a network based on both instantaneous network status and user defined policies [4]. This results in optimizing network configurations and improving network performance. The main benefit of SDN is further evidenced by the fact that SDN offers a convenient and reliable platform for new techniques and motivate new network designs, attributed to its network programmability and the ability to define isolated virtual networks by means of the control plane.

The communication between the centralized control plane and the data plane is extensively delineated by OpenFlow. The OpenFlow-based SDN has innumerable merits, SDN network integrated with legacy traditional networks is difficult to organize [5]. For example, even though OpenFlow-based SDN may exist amidst other legacy traditional protocols, an administrator of the network and operator wants to integrate them and configure flexible and simple network environment taking SDN advantages. Hence researchers have proposed innumerable solutions and have attempted various researches over the SDN.

2. SDN ARCHITECTURAL DESIGN

SDN has a High level reference architecture components and also faces some potential challenges for implementation. SDN and virtualization have the promise in simplifying network control operation by adding management flexibility, thus allowing the rapid development of new service offerings by enabling programmatic and centralized control of transport networks and network accessories like router and switches. The structure and function of traditional network has become increasingly complex.

Figure 1 provides various types of Transport SDN components and a brief illustration of the SDN architecture.

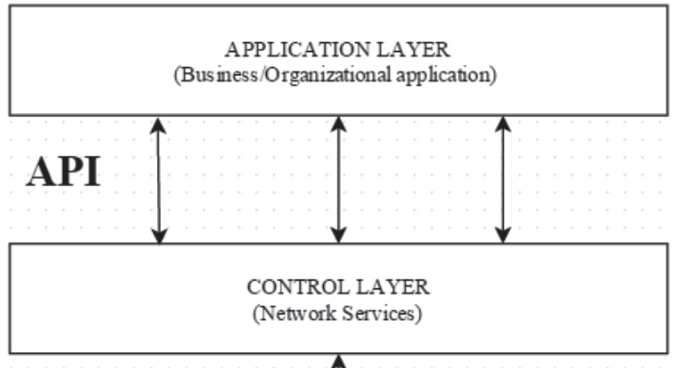

Figure 1: SDN Architecture

This architecture does not concentrate only at a specific set of protocols or specific hardware and software implementations. The modus operandi for thriving SDN ecosystem is by the separation of management, control and transport planes by the implementation of well-defined interfaces and standardized protocols to reach the desired level of interoperability between network components, vendor implementations, carrier network domains, and Data Center functions [6]. This technology also be applicable for packet as well as for circuit switching technologies.

A) Application Plane, Data Plane and Control Plane: The application plane is also known as the application layer, it provides a basic set of services and applications such as an intrusion detection system (IDS), intrusion prevention system (IPS), deep packet inspection (DPI), load balancers, security monitoring, and access controls. Application plane also act as the interface between the user and control plane which results in simplified operation of the System [7]. In SDN, the control plane is made to operate in a centralized approach. An SDN controller is able to access only a significant small part of the whole network, known as the SDN domain. Data centers, predominantly those in the cloud [8], are a logical example of an SDN domain. By employing SDN more often, exchange of information between multiple SDN domains will become a dominant need. A network can have various SDN domains, each supervised by an individual SDN Controller.

Interconnecting these controllers to share information and coordinate their decisions is principal key for routing information and deliver the quality in the service. In the future, when SDN will be made to employ in large-scale networks, operators of this large scale enterprise will want to divide the entire network into multiple connected SDN domains for more desirable scalability and security of the system [9].

B) Centralized Controller and Virtualization: As far as the SDN system is concerned, the network configuration protocols are governed by the centralized SDN controller on the basis of the familiar network topology. In a real-time environment and dynamic, namely the network for Smart Grid, it is essential for the SDN controller to be able to rapidly adjust to network and topology variations, due, for example, to the reconfiguration of the distribution grid. The adaptive SDN controller must be able to transform its decisions on the basis of the real-time performance of the network infrastructure [10]. The forwarding strategies of the adaptive centralized SDN controller takes the responsibility for account static configurations, like the topology or topology changes, as well as variations in QoS propounded by a link. SDN will reinforce IoT by centralizing control, generalizing network devices, and allowing flexibility, dynamic, automated reconfiguration of the networks. SDN allows the wide-ranging facility for network provisioning with predefined policies for plug-and-play set-up of IoT devices, automatic detection and overcoming of security threats in the network, and the providing the edge computing and analytics environments that turn data into insights. Centralization of control and virtualization of data by employing a software that has entire information regarding the network, enabling automated, policy-based control of even very large or a complex networks. Given the huge potential scale of IoT environments, SDN is plays an important role in making them simple to manage and maintain. A network virtualization hypervisor for Software Defined Networking (SDN) is the principle key component for the implementation of virtual SDN networks (vSDNs). Virtualizing software defined networks enables users to get their own SDN controllers in order to individually program the network control of their virtual SDN networks. A hypervisor plays as a barrier layer between the user SDN controllers and their respective virtual SDN networks. The hypervisor is composed of the network functions that are obligatory for virtualization, for example, translation or isolation functions. For scalability, the hypervisor can be realized through a multiple physically distributed instances each hosting the required virtualization functions [11]. Hence the physical locations of the instances, which realize the hypervisor, may create an impression on the overall performance of the virtual SDN networks. Network virtualization allows simplification to the general SDN controller placement problem.

C) OpenFlow: The researchers started gaining interest in the idea of SDN and started tackling all the loopholes and the started working on the drawbacks faced by SDN, also these activities were encouraged by the success of experimental infrastructures. All the researches underlying SDN faced lot of rigidity between fully programmable networks and pragmatism that would enable real-world deployment before the emergence of the OpenFlow. OpenFlow was successful in creating the balance between the primary two goals by enforcing multiple functionality than the traditional route controller and constructing on existing switch hardware [12]. One of the primary characteristic of SDN is the separation of the forwarding layer and its respective control layer and the use of a standardized protocol in between them, hence allowing the network control to operate on a commodity compute engine. This ideal pattern avoids the obstacle of the proprietary implementations of routers and allows interoperability between forwarding plane and control planes given by unique vendors. SDN supports for greater programmability which in turn facilitates for new etiquette which are not considered by standards or vendor implementations. OpenFlow is the first standard which acts as an interface for the communication between the control and forwarding layers of an SDN system [13]. OpenFlow allows direct access to forwarding and control plane and also plays a primary role in manipulation of the forwarding plane of network devices like routers and the switches. OpenFlow-based SDN technologies allows organization to address the high-bandwidth, dynamic nature of real time applications, modify the network to altering business needs, and significantly diminish operations and management overhead.

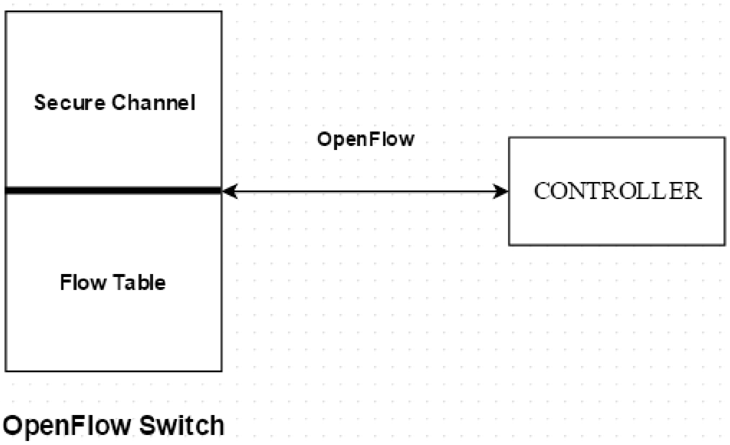

Figure 2: OpenFlow Switch

In SDN, a centralized controller is employed to be controlled by a programmable switch of Active Networking. On the other hand, OpenFlow is bounded to the fact that of less programmability than Active Networking because programmer of OpenFlow can select some action kinds assembled in advance in the OpenFlow specification. In addition after the switch finds a flow by monitoring the flow table, the actions are executed one by one in a sequence manner [14]. One of the primary advantage of SDN is that the most modern Ethernet switches and routers makes use of flow tables for important networking functions in the system, such as routing, firewall protection, subnetting and statistical analysis of data streams in the network. In an OpenFlow, the protocol for the communication mainly employs three parts in the entry of the flow table they are, “header” which is used for matching of the received packets in the network, “action” which is mainly used to define the decision to be taken for the matched packets, and “statistics” of the matched traffic flow. The OpenFlow protocol employs appropriate flow table handling services for a controller to insert, delete, modify, and lookup the flow table entries through a secure TCP channel remotely from some other location.

3. PERFORMANCE STATISTICAL ANALYSIS

As the communication network is rapidly increasing in an exponential manner, the number of network devices used in the network also increase, which is proportional to the complexity induced in the network. One of the main disadvantage of the SDN is that the added functionality and flexibility demands additional overhead on the equipment and as a result the network faces some penalties as far as the performance of the network is concerned [15][16]. This does not indicate that the overall performance of the network is decreasing [17] [18], SDN supported network is simple and performs the task in quicker way when compared to the traditional network design. We verified the performance of the SDN network by utilizing a dedicated bandwidth for communication between the control and data plane.

A) Methodology: We analyzed the performance of the SDN network which employs the dedicated bandwidth with the traditional SDN network which is made to run in a user space of the Linux. Dedicated bandwidth is designed which has two switches and single centralized controller by employing python programming which is made to run in the user space of the Linux.

In order to analyze the performance between these two types of design. To test and compare the throughput, delay and jitter among these two design, we observed the TCP/IP flow from one switch to another switch for both the design, and the corresponding results are plotted and the variations on the flow of the TCP/IP were examined for these two designs.

B) Experimental Testbed: The SDN's and the dedicated systems implemented on the Unit under test (UUT) is based on the Intel core2Duo e6600 CPU which is running at 3GHz, 4 GB of DDR2 memory and two Intel Intel Gigabit NIC's.

Throughput test is carried out in the Testbed by observing the TCP/IP packets between the two networks in the system for different workloads.

C) Results: This section provides the reports which describe the difference between the traditional SDN performance with the SDN design which has employed the dedicated link between the control and the data plane.

i) Delay Comparison: We can conclude from the graph (Figure: 3) that the SDN network with a dedicated link outperforms the traditional SDN network as far as the delay is concerned. Performance of the system is improved since the delay is inverse function of the performance of the system

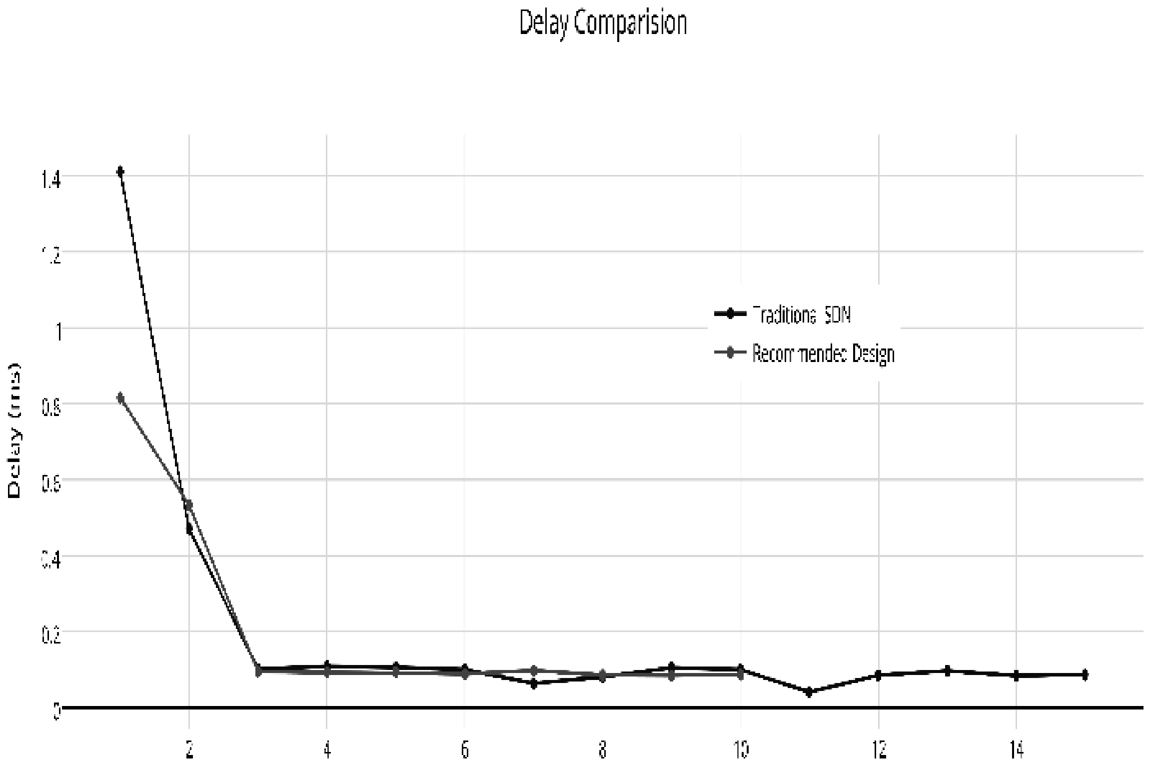

Figure 3: Delay Comparison

ii) Throughput Comparison: We can infer from the plot (Figure 4) that there is and improved performance in terms of the throughput of the system when both the designs are compared. Although the recommended design is more complex, its performance as the function of throughput is higher than the traditional design which is less complex.

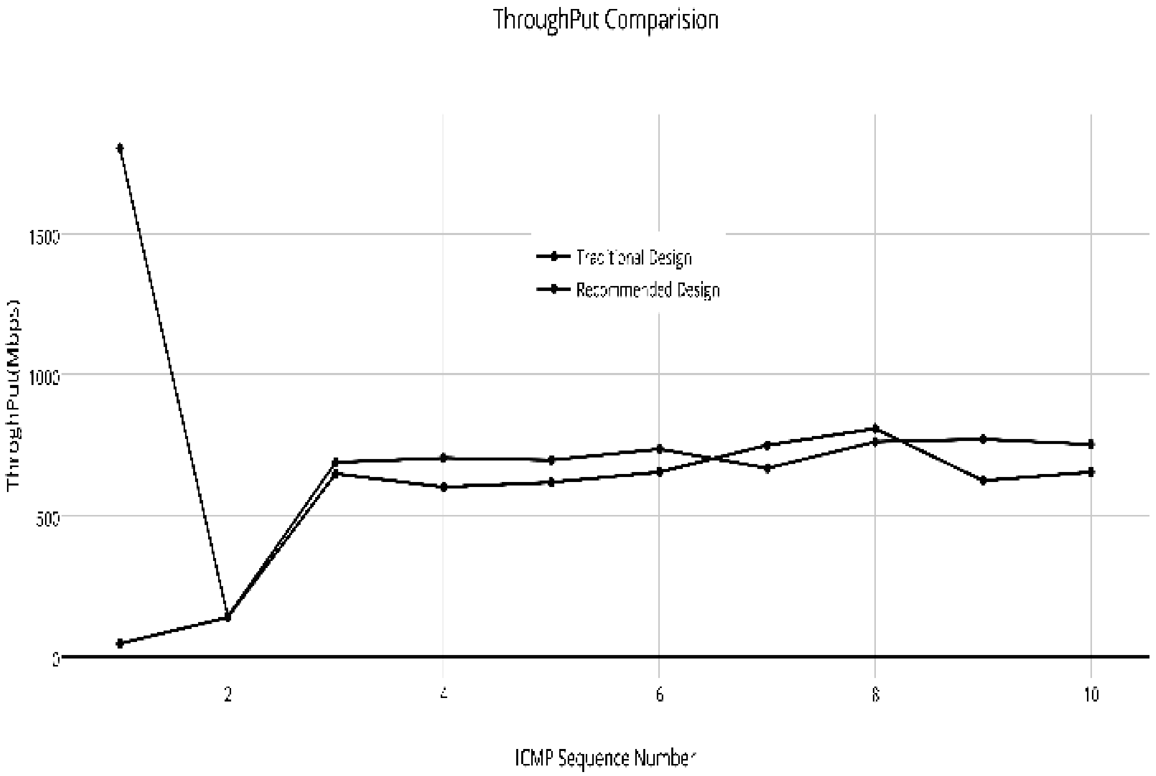

Figure 4: Throughput Comparison

iii) Latency Comparison: The Figure 5 shows the comparison results of the two designs as far as latency is concerned. It is evident that the recommended design outperforms the traditional SDN, which shows higher latency. From this result we can also infer that the complexity of the SDN architecture does not always influence the performance of the Network.

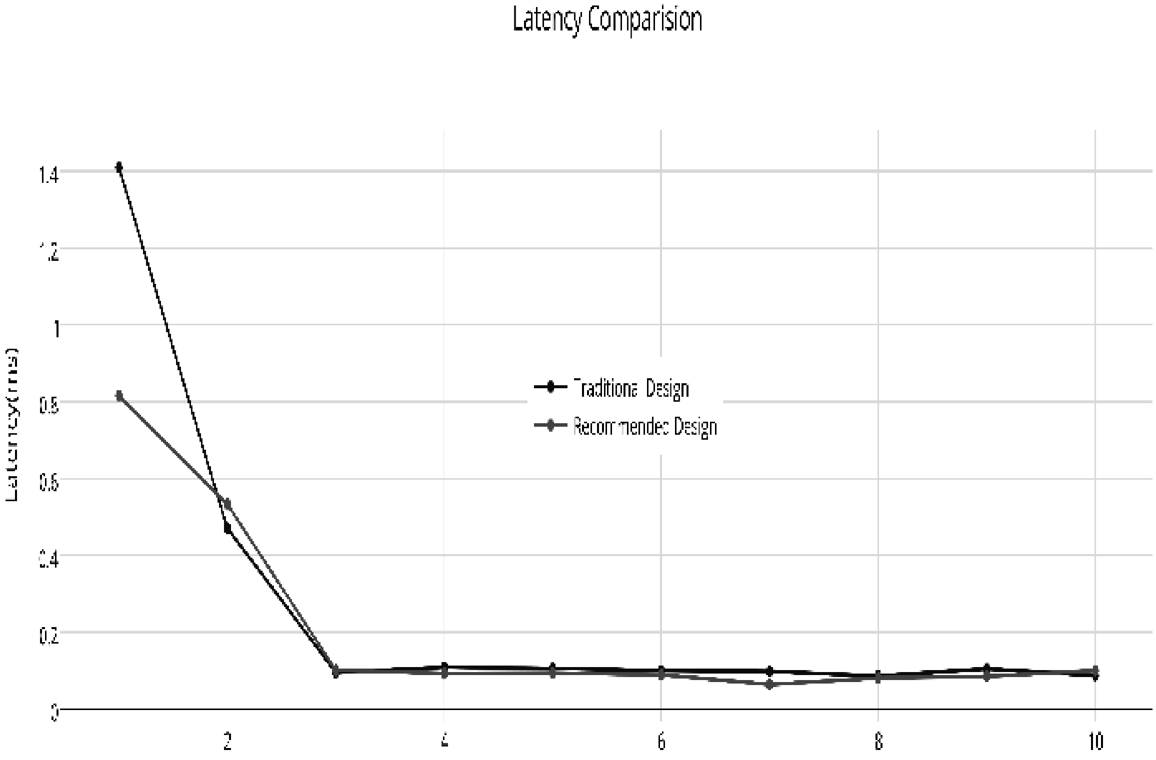

Figure 5: Latency Comparison

4. SUMMARY AND CONCLUSION

In this paper we have analyzed the performance of both the SDN architecture, which has varying characteristics as a function of functionality, flexibility and complexity of the SDN. We can perceive that the SDN offers desirable performance when employed with a separate dedicated link for the control and data plane. We observe that the SDN gives much desirable performance in terms of delay, jitter and the throughput of network. But the network complexity increases as a function of these changes made in the network design. Hence the future work of this paper involves in optimizing the complexity induced in the SDN.

Reference

1. Xia, W., Wen, Y., Member, S., Heng Foh, C., Niyato, D., & Xie, H. (2015). A Survey on Software-Defined Networking. *Ieee Communication Surveys & Tutorials*, *17*(1), 27–51. doi:10.1109/COMST.2014.2330903
2. Open Networking Foundation. (2012). Software-Defined Networking: The New Norm for Networks [white paper]. *ONF White Paper*, 1–12.
3. White, O. N. F., & January, P. (2015). OpenFlow Conformance Test Program. Retrieved from https://www.opennetworking.org/images/stories/downloads/working-groups/openflow-conformance-test-program-wp.pdf
4. *Andersson, J., Andersson, J., & Termander, E. (n.d.). Performance Management in Software Defned Performance Networking Management in Software Defined Networking, (November 2014).*
5. *Shukla, V. S. (2015). SDN Transport Architecture and Challenges. Optical Fiber Communication Conference, 02030, W4J.1. doi:10.1364/OFC.2015.W4J.1*
6. Hata, H. (2013). A study of requirements for SDN switch platform. ISPACS 2013 - 2013 International Symposium on Intelligent Signal Processing and Communication Systems, 79–84. doi:10.1109/ISPACS.2013.6704525
7. K. Co and S. Korea, "*An Architecture for SDN Flowmap Inter-operation with Legacy Protocols*," pp. 135–137, 2014
8. S. Koizumi and M. Fujiwaka, "*SDN + cloud integrated control with statistical analysis and discrete event simulation*," pp. 289–294, 2015.
9. Ieee, C. E. Rothenberg, M. Ieee, S. Azodolmolky, S. M. Ieee, S. Uhlig, and M. Ieee, "*Software-Defined Networking : A Comprehensive Survey*," Proc. IEEE, Vol. 103, No. 1, pp. 14 – 76, 2015.
10. Feamster, N., Rexford, J., & Zegura, E. (2014). The Road to SDN: An Intellectual History of Programmable Networks. *ACM Sigcomm Computer Communication*, *44*(2), 87–98. doi:10.1145/2602204.2602219
11. T. Wood, K. K. Ramakrishnan, J. Hwang, G. Liu, and W. Zhang, "Toward a Software-Based Network: Integrating Software Defined Networking and Network Function Virtualization," no. June, pp. 36–41, 2015.
12. Y. Jimenez, C. Cervello-Pastor, and A. Garcia, "Dynamic Resource Discovery Protocol for Software Defined Networks," IEEE Commun. Lett., vol. 19, no. 5, pp. 743–746, 2015
13. Y. Zhang, Y. Tang, D. Tang, and W. Wang, "*QOF: QoS Framework Based On OpenFlow*," pp. 380–387, 2015.
14. R. Casellas, R. Muñoz, R. Martínez, R. Vilalta, L. Liu, T. Tsuritani, I. Morita, V. López, O. G. De Dios, and J. P. Fernández-palacios, "*SDN Orchestration of OpenFlow and GMPLS Flexi-Grid Networks With a Stateful Hierarchical PCE [Invited]*," Vol. 7, No. 1, pp. 106–117, 2015.
15. Gelberger, A., Yemini, N., & Giladi, R. (2013). Performance Analysis of Software-Defined Networking (SDN). 2013 IEEE 21st International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems, 389–393. doi:10.1109/MASCOTS.2013.58
16. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, and R. Sherwood, "*On controller performance in software-defined networks*," in Proc. USENIX Hot-ICE, 2012
17. Yuta Watanabe, Yoshitaka Nakamura, Osamu Takahashi, "*A method to improve network performance of Proxy Mobile IPv6*" presented at 2015 Eighth International Conference on Mobile Computing and Ubiquitous Networking (ICMU).
18. Sallahi, A., & St-hilaire, M. (2015). Optimal Model for the Controller Placement Problem in Software Defined Networks. IEEE Communications Letters, 19(1), 30–33. doi:10.1109/LCOMM.2014.2371014.