Indian Journal of Science and Technology, *Vol 10(29), DOI: 10.17485/ijst/2017/v10i29/112447, August 2017*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Software Defined Networking: Research Issues, Challenges and Opportunities

Shailendra Mishra[1]* and Mohammed Abdul Rahman AlShehri[2]

[1]Department of Computer Engineering, College of Computer & Information, Majmaah University, Saudi Arabia; s.mishra@mu.edu.sa
[2]Department of Information Technology, College of Computer & Information Sciences, Majmaah University, Saudi Arabia

Abstract

Objectives: This paper focuses on challenges, opportunities and research issues of software defined networking (SDN), as well as how to select the best possible SDN controller, which in result will help to reduce the complexity of a network, price of implementation and maintenance of the network in any big organization. **Methods/Statistical Analysis:** In order to meet the objective, the review of literature has been carried out in the following contexts; Software defined networking, SDN protocol (Open Flow) and SDN research challenges. Software defined networking is one of the most discussed topic these days. This technology is being considered one of the favorable technologies for isolation of control plane and data plane and logical placement of centralized control from SDN controller. This research focuses on major issues, challenges and current requirements of network implemented in any big organization where traditional network is being implemented. **Findings:** To solve the issues of multiple located branch networks, cost, technical resources at each location, expertise, separate control plane for configurations, decentralized visibility of network devices, separate VLANs for each branch, complex traffic engineering, limited physical access of branches w.r.t working hours, bandwidth bottleneck at each branch, we surveyed literatures and web resources for the existing SDN controllers like NOX, POX, Ryu, Floodlight, and Open Daylight etc. All these controllers are based on Open Flow protocol. The primary concern is that a conventional system develops gradually, it has a generally abnormal state of operational expenditure and moderately static in nature. SDN holds the guarantee of overcoming those confinements. Major issues which are being faced are increasing requirements from user side, bandwidth availability, hardware (switches requirement at every place), technical resources are required at remote site for configurations, scalability issues, cost, high level processing power at each device, traffic engineering, resiliency against failures, decentralized visibility of hardware devices etc. SDN will helps to improve centralized visibility as all the underlying open flow switches are connected to controller, all switches can be configured from SDN controller without accessing individual switches. Research papers referred in this paper provide a bird eye view of what may cause hurdles further in development and technology integration of technology. **Application/Improvements:** This research will help how to select the best possible SDN controller which in result will help scalability, less hardware and software requirements, less technical resources requirements, centralized visibility, hassle free traffic engineering and high availability of network.

Keywords: AHP, MCDM, Openflow, SDN, SDN Controller

1. Introduction

In the recent years, the hot topic in network administration has been Software Defined Networking (SDN) and SDN controllers. Software Defined Networking has come up with the solution of traditional network issues which includes separate control plane for each network device in traditional network including switches and routers, so that not only the decision is made on each device but also the processing takes place at each device. With traditional

*Author for correspondence

network, there is no centralized visibility of network devices and the needs to be installed Network Monitoring System (NMS) for full visibility.

To better comprehend why SDN has turned out to be so vital? We need to take a glance at what existed before SDN. Customary systems administration structures have critical constraints that must be overcome to meet present day IT prerequisites. Today's system must scale to suit expanded workloads with more noteworthy spryness, while likewise keeping costs at least. In any case, the conventional approach has significant restrictions. At first, there was a considerable measure of buildup around SDN preceding comprehension genuine client utilize cases.[1] In the past few years SDN attracted many industries due to a lot of advantages mainly due to network virtualization.[2] The main characteristic of SDN, it combines the control & data plane, this way a single software control program manages and controls the various data plane elements.[1] Customarily, associations had a straightforward response to developing interest for information limit and expanding data transmission needs, as well as limit the cost of costly equipment's. Shockingly, most companies can no longer manage the cost of such an exorbitant approach, particularly notwithstanding the exponential development sought after. SDN is also utilizing server virtualization to increment asset effectiveness, improve the manual IT procedures and tune applications and systems.[3] SDN makes utilization of virtualization to incredibly grow, organize proficiency and accordingly give answers for the need to expand the limit without burning up all available resources, and disentangling the administration of those united assets.[3,4] SDN provides the solution to the issues of the traditional networking, which includes separate control plane for each network device in traditional network devices like router and switches, so that the decision is made on each device and it takes processing at each device.[4]

In the case of Virtual Local Area Network (VLAN), it is scalable up to 4096 as VLAN bit in Ethernet frame is 12 bit which comprises of 4096 combinations.[5] Therefore, there can be as many as 4096 networks in a network which minimize the scalability of growth in a network. Traditional network restricts the scalability of network expansion. Access to each device separately, configurations on each device separately makes harder for work to configure each device.

Protocols running on devices as proprietary from companies which is not giving permission to change on source code.[5] No open programmable interface for devices in traditional network. As in traditional network there is no option for that processing of packets at each device for decision making such as routing or switching, so it takes CPU usage and delay in overall traffic, adding extra time for reachability of packets. With traditional Network, there is no centralized visibility of network devices and there needs to be separate Network monitoring system to be installed for full visibility.[4]In order to resolve these issues, SDN comes up with centralized approach with separate controller and the data plane, visibility of all networking devices, programmable interfaces, scalable, speedy network, OS compatibility as open interface, source code access.[2,6] Both data plane and the control plane exist inside the controller but they are decoupled for their work.[6] Data plane is used for carrying user traffic and control plane is used for communication between controller and network application for decision for forwarding the decisions and actions.

Many SDN Controller platform has come into existence in the past few years, like Beacon OpenFlow controller, NOX, POX,Nettle,OpenDayLight, FloodLight, Ryu.[7-12] By using these platform researchers develop many applications such as load balancing,network virtualization, energy efficient networking, dynamic access control in enterprise network, Virtual machine mobility etc.[3,13-17]

NOX is a network control platform,it is an OpenFlow controller for development of management functions for organizations and home networks. POX is a python based SDN controller that is acquired from the NOX controller,OpenDayLight is an open source extend, also OpenFlow switches can handle mixed OpenFlow and non-openFlow networks.

In order to meet the objective, the review of literature has been carried out in the following contexts:

- Software defined networking
- SDN protocol: OpenFlow
- SDN research challenges

2. Software Defined Networking (SDN)

Software defined networking is one of the most discussed topic these days, keeping in mind the fast growing industry and meeting the requirments of end to end connectivity, it is one of the most reasearched topics these days. It is a networking model that brings a lot of new proficien-

cies and solves the problems of old networking models. Software Defined Network is designed on sepration of netwrok intelligence,packet switching devices and merging network intelligence in a centralized controller. This crontroller then acts as the main brain or main controller which is responsible for the decision routing which is placed at switches using protocols named as "OpenFlow". This technology is being considered one of the favorable technologies for isolation of control plane &data plane and logical placement of centralized control from SDN controller[5]. Data plane and control plane are decoupled for various purposes.The implementation of software defined networking in existing networks is considered as the most complex issue. Despite the achievements so far, implementation of SDN is still considered to be in a very delicate stage.[18] Many researchers had agreed on a central cause for delay in implementation of SDN, complexity caused by new technology. Ongoing research and industrial equipment deployment could resolve some of the complications for performance, scalability, security, and interoperability issues.[19] No matter how much SDN face challenges during implementation, the advantages for scalability and reliabilty of software defined network justifies the approach and suggest further experiments with enhancing data center management.

The another important research field inside the SDN people group is security. Overall security is connected to control systems, it provides a barrier amongst clients and ensure the system is secure against vindictive and undesirable interlopers. There are two levels of security ,first conventions like Secure Socket Layer (SSL) and second one is, to check outer strategies to ensure organization's security.The primary concerened of second level of security is how to secure switches, servers and end devices from threats and attack. Inside SDN, second level of security assumes an essential part.[20] The OpenFlow term gives a tool to secure this composition.[12] It is up to the controller to secure the composed solution with OpenFlow switches and various controller implementations do not realize these security tools. At the point when no accumulation security is dedicated, a malicious device can imitate the controller and take control of the switches.[21] Consolidated activity can be redefined for inspection and data extraction. Applying join security between the control and information layer is therefore the primary part which must be satisfied to guarantee efficiency on the system.[22]

Various cases are available to show the vulnerabilities to the system. Invasive programming can meddle the system and data, additionally saturating attacks can weaken servers or over-burden OpenFlow switches and controllers. Security systems must be realized on the system to recognize malicious data flow and take essential activities to be genuine and reroute this data flow.[23] In current systems administration statesecurity concerened is at top priority.[23] Switches and firewalls perform security at layer 3, where hosts and servers have security applications at layer 7,with SDN, the possibility to apply security arrangements to all layers in systems administration.[23,24] The SDNarchitecture has three main layers, namely application layer, control layer and underlying infrastructure layer. Unlike traditional network architecture where each device has a separate control plane, in SDN architecture it is separated and centralized on an isolated process (called controller) running at control layer.This isolated process (controller) provides universal view of the network. Consequently, the applications and services running at application layer seems to be running on a solo, logical network switch. As the infrastructure layer becomes vendor-independent with SDN,infrastructure layer devices should be designed to recognize instructions from SDN controllers only,this makes the formation process easy and fast.[25] To reduce the risk of single point of failure,researchers in the paper suggested that for good network operation and maintanence do not proceed without proper back up of network operation, which requires a detailed migration plan and a well-defined method of activity procedure.Still, this approach/technology has not yet been developed, so that existing operators are faced with high difficulty instead of easy management of network architecture.[3,26,27] In the paper authors suggested that software defined networking architecture is not an easily transferable technology as compared to smaller networks, where development and deployment can be done without a skilled staff.[19] However in software defined netwrok serious deployment concerns for enterprices working on large scale data networks are required. Therefore unforeseen communication with other deployed networks can cause an higher datarate of the broadcast traffic produced from non OpenFlow friendly devices.

2.1 Openflow

OpenFlow system initially created at Stanford University now under dynamic gauges improvement through the Open Networking Foundation (ONF). Open Networking foundation(ONF) defined that OpenFlow protocol is based on SDN layered architecture, it is in between control plane

and forwarding plane as communication protocol.[28] Using OpenFlow, we can orchestrate physical network and virtual devices. OpenFlow is usually implemented between SDN controller and OpenFlow enabled switches and it uses flow tables for matching the traffic or flow going through the network. There are two types of OpenFlow switches, OpenFlow only switches, it uses only OpenFlow Operations and OpenFlow Hybrid Switch it uses both OpenFlow and normal Ethernet operations.[28] OpenFlow is made as an tool for analysts to rapidly and adaptably try different things with new SDN views and devices in a wide-range environment. It cuts up physical systems through a intermediation layer. The controller goes about as a straightforward transition between a system of OpenFlow switches and other standard OpenFlow controllers, and manages data transfer capacity, CPU use and fill tables. While OpenFlow has been referred ongoing situations around the world, still industry is not really prepared.[28]

For instance, it does not have an automatic route interface or web-based organization, so clients must manage policy registers to push changes out.OpenFlow controllers come in numerous varieties and all have a similar objective of controlling and arranging switches. In the paper, authors considersome controllers, all are OpenFlow agreeable, differences were found in programming syntax and support for various OpenFlow versions,well recognized implementationsimilar to NOX and the Open vSwitch (OVS) utilize the C/C++ dialect, where POX and Ryu are Python based controllers and Java based controllers are found in FloodLight and OpenDayLight.Ryu, OpenDayLight and informal ported adaptations from NOX supported OpenFlow ver. 1.3.[7-9] Both FloodLight and OpenDayLight offer web program based design,NOX, POX and Ryu share a comparative building structure and this shared structure is utilized to give a non specific outline of an OpenFlow controller. The controllers discussed abovecan supply the core application with interpretation modules for OpenFlow convention adaptation.

Basically OpenFlow characterizes a variation through which a constantly incorporated controller can control an OpenFlow switch. Each OpenFlow-agreeable switch keeps up one or more accumulation tables, which are utilized to perform module queries.Floodlight has been tried with both physical and virtual OpenFlow-based switches.Similarly,supports systems where integration of OpenFlow-based switches are associated through traditional and non-OpenFlow switches.

Traditional systems i.e. non-SDN(e.g. the present Internet and its administrations like web scanning, media spouting) don't offer a (dynamic) approach to express the full scope of client requirements, for instance throughput, delay, delay variety or accessibility.[4] Packet headers can translate packets, yet vendors ordinarily don't trust client procedures. Consequently a few systems attempt to interpret the clients requirements seperatly (e.g. through trasnfer analysis), which may bring about extra cost and at times prompts to misclassification. SDN offers the capacity for a client to completely indicate its needs with regards to a trusted relationship that can be adapted. Control choices are made on a method global perspective of the system state, instead of distributed in separate module at every system call. With SDN, the control plane goes about as a private, sensibly unified framework regarding both planning and determining asset clashes, and in addition selecting low-level device components,

In the paperauthors discussed that network can be controlled by programming interface called APIs.[29] Hardware independent technology is likely to be achieved by this technique. Insoftware defined network, devices such as router and switches are only forwarding the packets and all decisions are made by the controller which uses network applications inside controller which decides which actions should be taken for certain flow. Using SDN, network devices are configured centrally without have to configure individually.[30] In the paper authors discussed that Openflow is ascending protocol, for its use in SDN as Application Programming Interface (APIs), and it can run without the dependency for vendor for compatibility for any application/services.[31] Overall expense for network is reduced by using SDN with OpenFlow Protocol . In the paper authors found that SDN OpenFlow design enables accessibility of all network devices using APIs.[32] These APIs can get information about all the network working services like routing, QoS etc. In traditional network every network device needs to be provisioned by accessing it ,however here all the network devices are being provisioned from centralized location .

In the paper authors categorized two types of interfaces in SDN controllers, which has separate function.[33] First North-bound interface, it is known as application layer connected interface which communicates with the upper layer for getting updates about rules which has been to be implemented for communication. This is high level communication. Second south-bound interface, it is usually for downward communication with the network

status updates about forwarding policies being pushed to downward devices which are usually switches,generally this is called infrastructure layer.

In the paper authors discussed that in infrastructure network there can be requirement of more than one controller for which communication is required between controllers for high-availability,communication is done between controller for network status sharing and any forwarding decision being made on upper layer.[34]

In the paper authors did a comparison of several OpenFlow controllers(Ryu, NOX/POX, FloodLight, Opendaylight etc).[35] According to the comparison done in the paper, NOX doesn't provide service addition and chaning, no load balancing, no policy enforcement, no dynamic network traps, no routing and only partial network monitoring as compared to moden SDN controllers like Ryu, FloodLight, OpenDayLight .Ryu is a component-based SDN framework, that supports a appropriate platform for SDN applications to run on the top of Ryu controller.[35] It is an open source tool programmed in Python that offers well defined APIs and packet libraries and supports all versions of Open Fast Path (OFP) data plane stack. It is well verified with various OpenFlow switches and is properly compatible on Open vSwitch.

NOX is an OpenFlow controller ,itis built in C++ or Python programming language and it provides a high-level programmatic interface. The recent version comprises of set of sample applications and some built-in libraries which provide useful network functions for instance host tracking and routing.[36] In the paper authors clarified POX as an open source controller for improvement of SDN applications.[37] POX controller offers an able approach to execute the OpenFlow convention which is the real correspondence convention between the controllers and the switches.

2.2 Current Research: Selection of SDN Controllers

In the paper authors analyzed three controllers namely NOX-MT,Maestro and Beacons and in the article authors analyzed five controllers namely NOX, POX, Ryu, Trema, FloodLight & OpenDaylight and use Analytic Hierarchy Process (AHP) for selecting the best controller.[38,39]Also discussed essential requirement of network i.e transport layer support and virtualization, these are achieved only through open source controller.As compared with author research there is missing element of user input, real time data,bandwidth requirement,delay requirement which is base of our study and literature review.

SDN is developed from various researches (particulary SANE and Ethane), NOX was firstly created at Nicira Networks. NOX was the main OpenFlow controller. Nicira gave NOX to the inspection group in 2008, For a developer, NOX provides asynchornous and fast I/O. It is targeting latest Linux technologies. It provides a C++ OpenFlow 1.0 API. It also includes trial components for discovery of topology, switch learning and network-wide switches.[40]

Floodlight controller is a SDN controller, it characterizes the open interchanges convention in a SDN domain, that permits the SDN controller (brains of the system) to address the sending plane (switches, routers etc.) . In the article author has discussed the multiple criteria decision making in his book, which we will use to model the SDN architecture in simple tabular form to find out its properties for the selection of best SDN controller to be used for a big organisation.[41]

SDN controllers have several different properties which affect the efficiency of the network. Selecting controller using single property is insignificant,therefore select multiple properties of the controller to find out their impact in the efficiency and effectiveness in network.One approach of selection process is dicussed in detail in the paper[41,] and the process is called Multi-Criteria Decision Making (MCDM).[41] In the article authors discussed the benefits & limitations of Analytic Hierarchy Process (AHP) for solving and getting the results from multiple properties.[42] AHP is used in helping decision maker facing complex problems having multiple subjective and conflicting choices. In the paper authors has given advice for using selection technique to select the best controller for two reasons, first it uses pairwise comparisonas well as consistency checking technique and secondly it allows decision makers to measure the relative significance of selected object.[43]

3. Modernization using SDN-based Network

Managing network devices and administrations can be integrated in SDN system applications, numerous system applications have been proposed by the SDN research group. They can be characterised as SDN Network Management, Load Balancing, SDN security, virtualization etc. administrations that are added to a SDN-based

system frequently require just a reasonable number of control components to be updated.[52]

3.1 SDN Network Management

Google depicts that inner worldwide system interconnects their server planes around the world. Server plans operations are relying upon the accessible data transfer and it is possible due server synchronization and correspondence using SDN and OpenFlow. This is conceivable in the server environment of Google. In the paper IP multicast is executed in the control plane by using OpenFlow protocol.[44] The control programming introduces the transfer segments in the changes as indicated by the multicast application. SDN arrangements support a programmable information plane, e.g FLARE and how to use interfaces for software defined networks is effectively discussed in the article.[45] The authors design and implemented a remote system administration interface to OpenFlow systems called OpenFlow Mangement Infrastructure (OMNI), it facilitates the administration of OpenFlow-based systems.[46] It can be troublesome because of the quantity of observing factors and various system setup choices. OMNI screens , arranges the dynamic creation of flow, it gathers information properties of the system devices and gives a straightforward system wide perception of the present system state.

3.2 Load Balancing for Application Servers

Different SDN-based applications have been proposed for big commercial systems.A typical application is load balancing for application servers, An OpenFlow switch consequently disseminates this activity for various servers, it is done by a centerlied device (load balancers),it recived the packets are specifically sent towards support counterfeits.[47]

3.3 Security and Network Access Control

SDN can control the flow of virtualized resourceses[48]. Service Function Chaining (SFC)(firewalls, load balancers etc) is responcibe for system security and access control ,SFC is discusses in detail in the paper.[49] As discussed in the article SDN can accomplish useful work for securing by managing activity to resourceses dependent upon system strategies.[45,50] In the paper,the authours propose the framework "Reverberation", that gives dynamically control approaches, because of the adaptability and the probability of fine-grained channels, they can accomplish receptive and dynamic system without the utilization of particular resourceses.[51] Similarly, the requirement of system wide management, is discussed about in the article, the authours proposed the OpenManaged design, which can be utilized to check and uphold control strategies and invariants continuously.[46,52] Such invariants can incorporate checks for sending loops, flawed instructioned and manage violation,depends on self-arranging maps to group activity designs.

4. Research Issues & Challenges

Major issues are;

1. How to make the subjective and conflicting choices using SDN Controllers for complex problem decision maker?
2. Identifying the properties of SDN Controllers and their impact on complex problem decision making technique.
3. Identification of best SDN controller using AHP technique.
4. SDN controller should be act like active-standby and in the case of one controller is down all the flow tables should go to backup controller, so that traffic operations are not disturbed.
5. Dynamic multipath load-balancing, in case of congestion load can be shared.
6. Handling huge amount of data in the network requires high CPU and memory at controller side so that all the requests are handled without being delayed.
7. Dependency of how the number of flows SDN controller can handle from open flow process which setup the flow process each time new flow is started.
8. Openflow hardware switches sends flow for decision making to SDN controller and in this way thousands of flow requests can come, which can make latency
9. Controller Placement is very important as it can cause delays if not proper placed.
10. Interoperability issue in SDN is one of the major issues. There is need to have interface for communication between SDN and non-SDN control plane like MPLS, this will increase the scalability of network being run.
11. From Security point of view, there should be some ACL Access control list.

5. Conclusion

By using SDN platform researchers develop many applications such as load balancing, network virtualization, energy efficient networking, dynamic access control in enterprise network, virtual machine mobility etc. To solve the issues of multiple located branch networks, cost, technical resources at each location, expertise, separate control plane for configurations, decentralized visibility of network devices, separate VLANs for each branch, complex traffic engineering, limited physical access of branches w.r.t working hours, bandwidth bottleneck at each branch, we surveyed literatures, web resources and books for the existing SDN controllers like NOX, POX, Ryu, Floodlight, and Open Daylight etc. All these controllers are based on Open Flow protocol.

SDN means that systems are controlled by programming applications.SDN is a design, indicating to be changing, sensible, savvy, versatile, trying to be accountable for the high data transmission, which is the dynamic in nature of applications nowadays. The OpenFlow convention can be used as a part of SDN advancements. The SDN engineering can be categorized as flexible i.e. separating control from data plane. Network insight is brought together in programming based software defined networking controllers, which keeps up a worldwide perspective of the system. It is automatically arranged i.e. system assets rapidly by means of dynamic and mechanized SDN programs.

The key benefits of a SDN technique is to gather all ranges of the association, as a component of server virtualization, Virtual Machines (VMs) are progressively moved between servers in a matter of seconds or minutes. Numerous SDN technologies are truly at the point for facilitating cloud computerization arrangements. This prompts to more noteworthy between operability, more development, and more adaptable, financially savvy arrangements. It could be controlled by numerous SDN controller applications, different SDN models are advancing in diverse ranges, and effective SDN techniques will consistently be in the market, between operable multi-dimensional communities with key open source innovations or institutionalized solutions.

6. Reference

1. Feamster N. The Road to SDN An intellectual history of programmable networks. Publisher ACM. 2013; 11(12):1–21.
2. What is SDN. Available from: www.sdxcentral.com/sdn/definitions/what-the-definition-of-software-defined-networking-sdn. Date accessed: 01/01/2017.
3. Patouni E, Merentitis A, Panagiotopoulos P, Glentis A. Network Virtualization Trends: Virtually Anything Is Possible by Connecting the Unconnected. IEEE SDN for Future Networks and Services; 2013. p. 1–7.
4. Traditional Vs Software Defined Networking. Available from: www.ipknowledge.net/wp-content/uploads/2014/12/SDN.pdf. Date accessed: 04/01/2017.
5. Traditional Network Infrastructure Model and Problems Associated with it. Available from: http://www.pluribusnetworks.com/traditional-network-infrastructure-model-and-problems-associated-with-it. Date accessed: 12/12/2016.
6. Astuto ABB. Nunes, Mendonca M, Nguyen XN, Obraczka K, Turletti T. A Survey of Software-Defined Networking Past Present and Future of Programmable Networks. IEEE Communications Surveys & Tutorials. 2014; 16(3):1617–34. Crossref
7. Kaur S, Singh J, Ghumman NS. Network Programmability using POX Controller. Proceedings of International Conference on Communication Computing & Systems (ICCCS). 2014; 1:134–8. PMid:25143894 PMCid:PMC4137641
8. Floodlight. Available from: http://www.projectfloodlight.org/floodlight/. Date accessed: 20/12/2016.
9. Lithium. Available from: https://www.opendaylight.org/lithium. Date accessed: 25/12/2016.
10. Erickson D. The Beacon OpenFlow controller. Proceedings of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN'13). Publisher ACM; 2013. p. 13–8. Crossref
11. Gude N, Koponen T, Pettit J, Pfaff B, Casado M, McKeown N, Shenker S. NOX Towards an operating system for networks. ACM SIGCOMM Computer Communication Review. 2008; 38(3):105–10. Crossref
12. Voellmy A, Hudak PN. Functional reactive programming of OpenFlow networks. Proceedings of workshop on Practical Aspects of Declarative Languages; 2011. p. 235–49. Crossref
13. Jarraya Y, Madi T, Debbabi M. A Survey and a Layered Taxonomy of Software-Defined Networking. Published in IEEE Communications Surveys & Tutorials. 2014; 16(4):1955–80. Crossref
14. Wang R, Butnariu D, Rexford J. Open Flow-based server load balancing. Proceedings of workshop on Hot-ICE-11 USA; 2011. p. 12.
15. Sherwood R, Gibb G, Yap KK, Appenzeller G, Casado M, McKeown N, Parulkar G. Can the production network be the testbed. Proceeding of 9th USENIX conference on Operating systems design and implementation. Vancouver Canada Publisher ACM; 2010. p. 365–78.

16. Heller B, Seetharaman S, Mahadevan P, Yiakoumis Y, Sharma P, Banerjee S, McKeown N. ElasticTree Saving energy in data center networks. Proceedings of the 7th USENIX conference on Networked systems design and implementation San Jose; 2010. p. 17–21.
17. Nayak A, Reimers A, Feamster N, Clark R. Resonance. Dynamic access control in enterprise networks. Proceedings of Workshop on Research on Enterprise Networking Barcelona Spain (WREN'09) Publisher ACM; 2009. p. 11–8.
18. Erickson D. A demonstration of virtual machine mobility in an Open Flow network. ACM SIGCOMM'08 ACM 978-1-60558-175-0/08/08. Publisher ACM; 2008. p. 1.
19. Caraguay ALV, Lopez LIB, Villalba LJG. Evolution and Challenges of Software Defined Networking. IEEE Communications Magazine; 2013. p. 1–7.
20. Mysore RN, Pamboris A, Farrington N, Huang N, Miri P, Radhakrishnan S, Vahda A. Portland a scalable fault-tolerant layer 2 data center network fabric. ACM SIGCOMM Computer Communication Review. 2009; 39(4):39–50.
21. Floodlightdocumentation. Available from: http://www.projectfloodlight.org/documentation/ Date accessed: 15/12/2016.
22. Heller B, Sherwood R, McKeown N. The controller placement problem, Proceedings of the First Workshop on Hot Topics in Software-Defined Networks (HotSDN'12) Helsinki Finland Publisher ACM; 2012. p. 7–12. Crossref, Crossref
23. Hunag Y, MinChi C, YaoTing T, YuChieh C, YanRen C. A Novel Design for Future on-Demand Service and Security. Proceedings of the 12th IEEE International Conference on Communication Technology (ICCT); 2010. p. 385–8.
24. Porras, Shin P, Yegneswaran S, Fong V, Tyson M, Gu M, A Security Enforcement Kernel for OpenFlow Networks. Proceedings of the ACM Workshop on Hot Topics in Software Defined Networks (HotSDN); Helsinki Finland; 2012. p. 121–6.
25. Implementation and Performance Analysis of Firewall on Open vSwitch. Available from: https://pdfs.semanticscholar.org/9bb1/4f91b269e72a98d9063d22da2092dc5f552d.pdf Date accessed: 11/11/2016.
26. Casado M, Koponen T, Shenker S, Tootoonchian A. Fabric A retrospective on evolving SDN. Proceedings of the ACM Workshop on Hot Topics in Software Defined Networks (HotSDN); Helsinki; 2012. p. 1–5. Crossref PMCid:PMC3389431
27. Lara A, Kolasani A, Ramamurthy B. Network Innovation using OpenFlow A Survey. IEEE Communications Surveys & Tutorials. 2014; 16(1):493–512. Crossref
28. Open Flow Switch Specification. Open Networking Foundation ONF. p. 106 Available from: https://wand.net.nz/~brad/papers/openflow-spec-v1.5.0.pdf. 19/12/2014.
29. Shin MK, Nam KH, Kim HJ. Software-Defined Network (SDN) A Reference Architecture and Open APIs in ICT Convergence. Proceeding of IEEE International Conference on ICT Convergence (ICTC); Jeju Island; 2012. p. 360–1.
30. Kim H, Feamster N. Improving Network Management with Software Defined Networking. IEEE Communications Magazine. 2013; 51(2):114–9. Crossref
31. Rothenberg EC, Nascimento RM, Salvador MR, Corrêa NAC, Lucena CS, Raszuk R. Revisiting routing control platforms with the eyes and muscles of software-defined networking. Proceedings of the ACM Workshop on Hot Topics in Software Defined Networks (HotSDN); 2012. p. 13–7. Crossref PMid:28083220 PMCid:PMC4945262
32. Bakshi K. Considerations for Software Defined Networking (SDN) Approaches and use cases. Proceedings of the mIEEE Aerospace Conference Big Sky MT; 2013. p. 1–9. Crossref
33. Xia W, Wen Y, Foh CH, Niyato D, Xie H. A Survey on Software-Defined Networking. IEEE communication surveys and tutorials. 2015; 17(1):27–51.
34. Xie H, Tsou T, Lopez D, Sidi R, Yin H, Aranda PA. Software-defined Networking Efforts. IETF Journal Published by the Internet Society in cooperation with the Internet Engineering Task Force; 2012.
35. SDN Series Part Eight Comparison of Open Source SDN Controllers. Available from: https://thenewstack.io/sdn-series-part-eight-comparison-of-open-source-sdn-controllers. Date accessed: 12/10/2016.
36. The NOX Controller. Available from: https://github.com/noxrepo/nox.Date accessed: 08/08/2016.
37. Fernandez, Marcial. Evaluating OpenFlow controller paradigms. Proceeding of The 3rd ACM SIGCOMM Workshop on Information-Centric Networking; 2013. p. 151–7.
38. Tootoonchian A, Gorbunov S, Ganjali Y, Casado M, Sherwood R. On Controller Performance in Software-defined networks. Proceeding of the 2nd USENIX conference on Hot Topics in Management of Internet Cloud and Enterprise Network and Services; 2012. p. 10.
39. Khondoker R, Zaalouk, Marx R. Feature-based Comparison and Selection of Software. World Congress on Computer Applications and Information Systems (WCCAIS-2014); 2014. p. 1–7. Crossref
40. Shalimov A, Zuikov D, Zimarina D, Pashkov V, Smeliansky R. Advanced study of SDN/OpenFlow controllers. Proceedings of the 9th central & eastern European software engineering conference in Russia Publisher ACM; 2013.
41. Gandibleux X. Multiple criteria optimization state of the art annotated bibliographic surveys. Springer Science & Business Media; 2006.
42. Ishizaka A, Labib A. Analytic hierarchy process and expert choice Benefits and limitation. Publisher Springer link. 2009; 22(4):201–20.

43. J David, D Saaty. Use analytic hierarchy process for project selection ASQ Six sigma forum magazine; 2007. p. 1–8.
44. Kotani D, Suzuki K, Shimonishi H. A Design and Implementation of Open Flow Controller handling IP Multicast with Fast Tree Switching. Proceedings of the IEEE/IPSJ International Symposium on Applications and the Internet (SAINT) Izmir Turkey; 2012. p. 60–7.
45. Bruno Astuto A. Nunes, Marc Mendonca, Xuan-Nam Nguyen, Katia Obraczka Thierry Turletti. A Survey of Software-Defined Networking Past Present and Future of Programmable Networks. IEEE Communications Surveys & Tutorials. 2014; 16(3):1617–34. Crossref
46. Mattos MFD, Fernandes NC, Costa TV, Cardoso PL, Campista MEM, Costa MKLH, Duarte OMNI. Open Flow Management Infrastructure. Proceedings of the 2011 IEEE International Conference on the Network of the Future; 2011. p. 52–6.
47. Reitblatt M, Foster N, Rexford J, Schlesinger C, Walker D. Abstractions for Network Update. Proceedings of the ACM SIGCOMM Conference on Applications Technologies Architectures and Protocols for Computer Communication Helsinki Finland; 2012. p. 323–34. Crossref, Crossref
48. Wang R, Butnariu D, Rexford J. Open Flow-Based Server Load Balancing Gone Wild. Proceedings of the USENIX Conference on Hot Topics in Management of Internet Cloud and Enterprise Networks and Services; 2011. p. 12.
49. Boucadair M, Jacquenet C. Service Function Chaining Framework &Architecture; 2014.
50. YuHunag C, MinChi T, YaoTing C, YuChieh C, YanRen C. A Novel Design for Future on-Demand Service and Security. Proceedings of the 2010 IEEE 12th International Conference on Communication Technology Publisher-IEEE; 2010. p. 385–8.
51. Erickson D. The Beacon OpenFlow controller. Proceedings of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN'13) Publisher-ACM; 2013. p. 13–8. Crossref
52. Yang M, Li Y, Jin D, Zeng L, Wu X, Vasilakos AV. SoftwareDefined and Virtualized Future Mobile and Wireless Networks A Survey. Springer. 2015; 20(1):4–18.