Software Defined Network: The Next Generation Internet Technology

Kshira Sagar Sahoo, Sambit Kumar Mishra, Sampa Sahoo, Bibhudatta Sahoo

National Institute of Technology, Rourkela, India - 769008

Abstract

Software Defined Networks is an emerging network paradigm which introduces programmability to networks and has the capability to dynamically configure the network. In a traditional IP based network the control part and the data forwarding elements are imposed in a single box that has very limited ability to configure the network, some vendor specific codes run on the forwarding elements to perform this task. SDN takes another approach by decoupling the controller part from the data plane part. It is the next generation Internet technology that solves the ossification of the Internet, along with creating massive innovations so that network management can be easily handled. In this paper, we are trying to explore the different layers, various protocols supported by SDN, along with discuss the numerous benefits of it.

Index Terms: SDN, Controller, OpenFlow, Control plane, Data Plane.

1. Introduction

In a traditional network, controller software and forwarding element, i.e., switches are integrated in a single box. Software instruction simply dictates on the forwarding element by imposing rules. These networks are complex and hard to manage .When we want to implement some new network policies the network operator has to configure each individual device separately by using low–level and other vender specific command. It becomes more complicated when both the control plane (which control the network traffic) and the data plane (i.e. Forwarding devices) are integrated into single network devices. This type of architecture reduces the flexibility and no chance for new innovation, so there is no elevation of network infrastructure.

Software defined network (SDN) [1] [2] is a new paradigm of next generation of Internet technology that separates the control plane from the data plane. The control part is taken away and is placed in a centralize location by means of the server is called controller. We realize this separation through a programming interface between the forwarding devices like switches and controller. The controller directly can have the command over the data plane elements through the programming interfaces (API). The most well-known API is OpenFlow [3], [4], [26].

1.1. Why software defined networks?

As we have discussed before, Software defined network separate the controller plane by decoupling from the data plane part. The early need of SDN was realized by the data centers when they find it difficult to manage their data center by using the traditional way of network management. Due to the cloud technology and burst of mobile device make it easy to access data from any corner of the world [15], [22] [24]. The unstructured data generated by social media, e-commerce site and many other sources cause data traffic on the network is increasing every day. To manage all these data, we need a new networking architecture which can dynamically manage the network. SDN fulfill all these promises it has capability to manage large network it do so by introducing, programmability and virtualization [20], [21] [25]. Now network function can realize as the service that can be accessed through the application. Network operators can write their own application program for the different purpose as per required. In SDN network comparable to the traditional network where the decision in the event was taken by the forwarding elements, now the forwarding elements are simply forwarding devices and all the control decision are taken by the controller. There are many challenges that traditional network facing can easily be solved in SDN network. For example the traffic engineering and dynamic load balancing was a challenging task in traditional network now in SDN it has separate module and application has been developed to handle this problem. For programmability of the network new programming language has been developed for different purposes, Pyretic is an example of such a language. Resiliency is one of the aspects of any network that recover a network from failover. SDN has its own approach to recover from network failure which makes it robust and available.

The contribution of this paper is summarized below:

- Discussed the traditional network architecture with SDN architecture.
- A deep insight into different layers along with support protocols by SDN.
- Finally, we have listed the benefits to the data centers by using SDN.

2. Traditional Network Architecture

It is important to overview the current network architecture to better understand the new changes proposed. Current network consists of transmission equipment connectivity between the components, software and communication protocols. For network connectivity, different medium connects, and those connections, different transmission modes exist. Such transmission modes are: Ethernet, wireless or optical. The core of any network is the hardware (routers and switches) that is connected together with one or more transmission methods.

When we consider the end to end communication, the information travels through the network equipment hope- by-hop until it arrives at its destination. Sometimes the information is segmented into multiple pieces and is sent through the different paths to get to its destination. Router and switches are devices that are stationed in the network to help in the network with the information that is being sent through the networks. They decide where and how the information will go from one point to another. These devices are also known as the network elements.

Deciding where the information should go is not a straightforward, in required knowledge about whom else is in the network. These types of information are managed by network protocols that mount in the network elements. The elements of control plane decide where the information is coming from, what type of information it is and how to reach the destination. Such protocols installed rule on a ternary content-addressable memory (TCAM) table.

Since we already know where the information goes, the next step is to know the how and where to send such information. In each of the network elements TCAM is the table lookup for the next hope of information of the particular instance. Once the next hope port is found, next it is just a matter of sending the information for the transmission. This is called the forwarding plane because the physical operation, it takes look for information and moves it to the next network element.

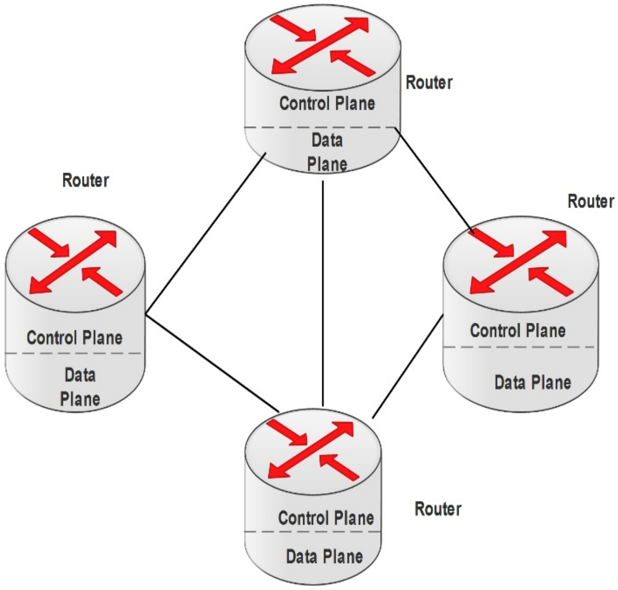

Fig 1. Traditional Network Architecture

2.1. Limitations of Traditional Networks

In traditional network both control plane and data plane embedded together in a single device as shown in fig 1. A typical router has a control plane that handles the routing information and a data plane that handles incoming packets that means where to send the incoming packets.

The network shown in the fig 1 comprises of only 4 routers. The amount of computation required for 4 router is easily imaginable, if we think of a larger network like WAN the number of devices are in thousands. In turn the delay values also increase when they share the routing information among themselves. Similarly, if one router fails the same routing decision will take by other router which takes more times. Apart from this some additional overhead like decoding packet header, forwarding data create extra burden to the traditional network. In this section we are discussing some limitations of the traditional network.

- Limited Innovation:

The one of the biggest issues that the current network architecture faces is that it is a closed system. Network equipment manufacturers have their own hardware, operating system, implementation of standards and their own extended set of feature. This means the new protocols cannot be tested and verified in a timely fashion. This is due to the fact that the manufacture has only access to the system.

- Different Management:

The operating system in such a system is close for the outside use. Beside the resource reservation protocols (RSVP) and its own management console, there is no other method to command the network equipment. The operating system does not expose access to outside command its internal system and this is a disadvantage because only two methods to manage it exist. When a large network has proprietary hardware from different vendors it makes it difficult to manage [12] [29].

- Costs

The equipment those are used for the network management are extremely expensive. This is due to the fact that the cost of the each element includes time spent on developing and verifying the protocols, which is a lengthy process. For example, it is impossible to get network equipment with only one protocol implemented by particular manufacturers. The whole OS is included for that particular model of hardware and its supported feature. This drive cost high.

3. SDN Architecture

In conventional network devices, both the control and data planes are embedded both in SDN, the control plane of a device moves to an external device called a controller. The routing decisions and routing tables are managed by the controller. The fig 2 shows a graphical representation of a typical SDN architecture. The routers receive incoming flows from the source and checks in its own flow table, if it doesn't exist, then it sends the first packet of the flow to the controller for taking the routing decision. After getting the packet the controller processes the header of the packet and updates the flow tables of the concerned device. The remaining packets of the flow are forwarded by the device instructed by the controller.

Though the router has the flow table, no longer it has the ability to modify the flow table, hence the router only become a forwarding device. Software defined networking can be seen as a set of different layer as shown in fig 3. In this architecture each layer has its own function. Following section introduces each layer in details. For each layer, the principal properties are explained based on different technology and solution.

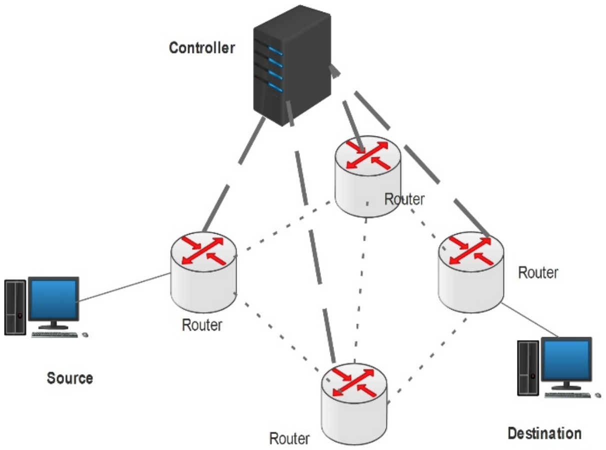

Fig 2. In SDN all the routers are managed under the controller

3.1. Layer I: Data plane

- Infrastructure Layer

The SDN architecture is similar to the traditional network which consists of set of networking equipment such as switches, router and middle box appliances etc. After introduction of SDN now the traditional devices are become data-forwarding devices. In SDN the network intelligence is taken away from the data plane elements called controller or NOS. Open standard interfaces are used to run these network such interface is OpenFlow.

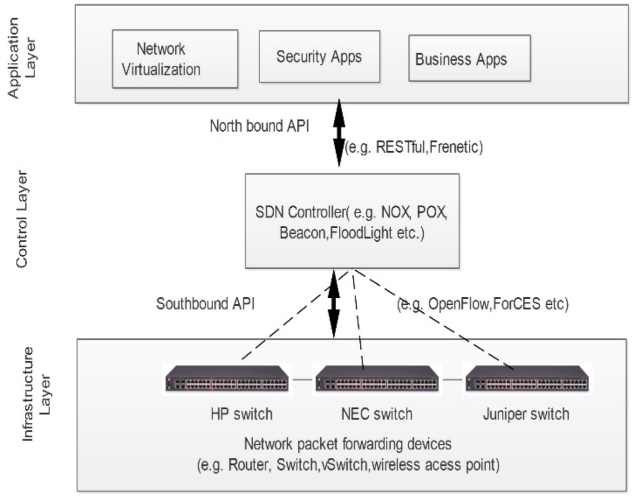

Fig 3. A typical SDN architecture

The data plane devices are most often hardware elements or sometimes software components dedicated in data packet forwarding. An OpenFlow application has the capability to enable these forwarding devices by introducing a number of flow tables. Each flow table contains the flow entries and each having three parts. These are 1) the matching rules; 2) the set of action against found packet 3) the counters that keep the statistics of matching packets.

- Southbound Interfaces

Southbound interfaces is the connecting channel between control plane and the forwarding elements. So this is the API that separates control and data plane functionality. Among few, OpenFlow is the most used southbound API for the SDN infrastructure. The OpenFlow protocols provide three information's to the NOSs. First the event based message is sent by forwarding devices to the controller when a link or port change is triggered. Second flow statistics are generated by the forwarding devices and collected by the controller. Third, packed-in message are sent by the forwarding devices to the controller when they do not know what to do with the new incoming flow.

Other than this, other southbound API for SDN are available such as POF, OpenStack, Open vSwtch, OVSDB, OpFlex, ForCES.

3.2. Layer II: Control plane

- Network Hypervisor

The fast research and development of the last decade has made the virtualization technology of a computing platform more popular. Hence, in the modern day computer, virtualization has already been an embedded technology.

In the modern cloud infrastructure, each user can have their own virtual resources [30]. This brings new opportunity for the service provider to build a new business model where users can demand for the service as per their requirement at the same time service provider can make better use of their available hardware. Unfortunately the power of virtualization technology has realized partially. Despite of the great advance in virtualization computing and the storage, the network configuration is still follow static arrangement. The static configuration occurs in a box by box manner. Different type of workload needs different type of network topologies and service. Today, virtualized workload has to operate in the same address space of physical infrastructure which is hard to achieve. To provide the complete virtualization, the network infrastructure should support arbitrary network topology and address scheme.

- Network Operating Systems/Controller

Legacy operating system provides a high level abstractions for accessing the lower level devices and have the capability to handle concurrent access to the under-lying resources. This functionality and the resource are the key elements for increasing the productivity, making the system and application development easier. On the contrary network is so far is manage by using some low level devices specific configuration mostly some close proprietary NOSs like juniper junOS and Cisco IOS. But all these hurdles can be easily handle by SDN and lessen the burden of solving networking problems by means of the centralize controller [28].

There are very diverse set of controller and controller platform have been developed and available in the market. These are differing by their architecture and design choice. Broadly it can be categories in its architectural point of view such as: centralized or distributed architecture.

In centralized architecture a single controller that manages all the connected forwarding elements in a network. So, there might be chances of single point of failure and scaling limitation. In a large network, to handle elephant flows a single controller is not sufficient. Centralized system such as NOX [5], Floodlight [6], Maestro and Beacon [7] are designed for the highly concurrent system. These controllers are designed based on multithreaded system to explore the parallelism on multi core computer architecture. In today's cloud and big data scenario we need a dedicated network, which can handle a huge volume and high speed data [14] [16] [23]. To manage all these sufficient numbers of controllers and the controllers should be in a suitable place in a network so that it will easier to manage the network.

This is the core part of SDN networks which operate between network devices and various applications at the user end. It has the responsibility to manage the flows by installing flow rules in the flow table of the network devices. In the Table 1 we are listing some controllers that support OpenFlow as the southbound protocols along with other north bound protocols [27].

Table 1: Some of the controller list

Name of the controller	Supported Technology for north bound and south bound
NOX	OpenFlow
Ryu	OpenFlow, Netconf, OF-config,
NodeFlow	OpenFlow
Trema	OpenFlow
MUL	OpenFlow
POX	OpenFlow
Opendaylight	OpenFlow, Netconf,
McNettle	OpenFlow
Beacon	OpenFlow

Table 2: Various open source network virtualization platform and design elements of SDN

Components	OpenDayLight	OpenContrail	HP VAN SDN	Onix	Beacon
Base network Service	Topology/stats/switches manager, shortest path forwarding	Tenet Isolation	Audit log, Alert, Topology, Discovery	Topology Discovery, Multi-Consistency storage	Topology , device manager, and routing
East/West bound APIs	_	Control node(XMPP-like control channel)	Sync APIs	Distributed I/O module	-
Integration Plug –in	OpenStack neutron	CloudStack, open Stack	OpenStack	_	_
Management Interface	GUI/CLI REST API	GUI/CLI	REST API shell/ GUI shell	_	Web
Northbound APIs	REST, Java APIs	REST APIs	REST APIs, GUI Shell	Onix API(general purpose)	APIs(Based on OpenFlow events)
Service Abstraction layer	Service Abstraction Layer(SAL)	_	Device Abstraction APIs	Network Information Base(NIB) graph with Import/ Export Function	_
Southbound APIs or Connectors	OpenFlow, OVSDV, SNMP,PCEP, BGP, NETCONF	_	OpenFlow, L3 agent, L2 agent	OpenFlow OVSDV	OpenFlow

Contrary to the centralized design of the controller, distributed NOS can be scaled up to meet the requirement any network environment, from small to large networks. Onix[8], HP VAN, HypeFlow[9] are the some controller have been design for the distributed SDN architecture

- Northbound Interface

For SDN a common northbound interface is still an open issue. This is a software system in which the implementation is mostly the driver. This interface is a software system of SDN which link the application interface with network operation system and provides an easy of abstraction to the network programmer. We can compare the northbound API with the POSIX system that granted of abstraction and application interface. The first north bound API was NOSIX. Which has the capability of higher level abstraction for northbound interfaces. Table 1.describes various controller's architecture and design elements.

Some of the controller such as Floodlight, NOX, and OpenDaylight proposed and define their own northbound controller APIs [18]. Some other proposal uses a different approach to allow applications to interact with the controller. The Yanc controller platform through virtual file system discover this idea by proposing a general controller platform based on LINUX system. As at present SDN use single Following section introduces each layer in details. For each layer, the principal properties are explained based on different technology and solution.

3.3. Layer III: Management Plane

- Programming Language

Programming languages have been used from the decades. Both in industries in academic practice languages evolves from the low-level hardware machine specific, such as assembly language to high level programming languages like Java and C++. The legacy networks is combinations of different types of heterogeneous devices like router and firewall and on the other hand network is responsible for various operations like load balancing, traffic monitoring, network control access etc.

Table 3. Programming Languages

Name	**Short description/ purpose**
FatTire	Use regular expression to program
Flog Chaning	Proving an event driven and forword
FlowLog	Provides a finite state language
FML	High level policy description language
Frenetic	Language design to avoid race condition
HFT	Enable hierarchal policies description
Maple	Provides a high-efficient multi core scheduler
Marline	provides mechanism for delegating management sub policies

The network management become a very complex task. To achieve these drawback SDN has introduced a simple solution for ease network management by using interface among the devices and the control software that controls the devices. As it has mentioned before OpenFlow is the well-known standard protocol which is used to redesigning the behavior of underlying devices. On the other hand, the SDN architecture require a high level abstraction to create various application.

The high- level programming languages can be intended to:

- Make a higher level abstraction for the forwarding devices.
- More innovative and productive environment can be designed by network software programmers.
- Modularization of software and code reusability in the control plane is another goal of high level programming language.
- Network virtualization can be more realized [17].

There are many challenges by using high-level programming language in SDN. The Table 3 provides information regarding some of the high level programming language.

4. OpenFlow

As we have stated before, OpenFlow is a standardize communication between the underlying devices and the controller in an SDN environment. The source code running on the top of the hardware devices cannot be altered. Hence, the network community was facing a lot of problem to test new designs in the current hardware architecture. As an experimental basis, OpenFlow was initially installed in campus networks and its goal is to provide a platform for researchers to experiment their work in a production network [10]. Most of the network switch industries have started implementing OpenFlow in their devices [19]. The Table 4 shows the list of several OpenFlow-enabled devices available in the market.

Table 4: Hardware companies using OpenFlow

Company	Model
HP	8200zl,6600
Extreme	Summit X670
IBM	Rackswitch G8264
Juniper	MX series
Pica8	P-3290

5. Benefits of SDN

Among many here a list of benefits are described.

- **Central management**: It provides a centralized view of the whole network that makes easier to centralize enterprise management and monitoring [13].
- **More coarse security:** Centralized security can be realized through Software Defined Network. The Controller provides a reliable central point of control to distribute security and policy information throughout the enterprise network [11].
- **Reduced Cost:** The overall operating costs will be reduced since many of the routine network administration issues can be programmed and centralized.
- **Less capital expenditure on hardware:** Since the intelligence of the data plane devices are moved to the controller in SDN, now they became white box switches. So, it is easier to optimize commoditized hardware.
- **Reduced the Downtime:** Device up gradation become easy in SDN, because it supports in virtualizing most of the physical networking devices [14].

6. Conclusion

We have gone through the detailed architecture and different layers of SDN. The specific advantages vary from network to network, but there are benefits in terms of network abstraction and more automation for the network administrator. Several organization like IETF, ONF, and ITU-T have started working on standardize the protocols of SDN aiming to provide better SDN solutions. Though SDN is the next generation Internet technology, various research challenges are still unexplored. In our future work we will focus and survey on various research challenges pertain to controller layer of SDN such as security challenges of controller, traffic engineering and controller placement problem.

References

[1] Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-defined networking: A comprehensive survey. Proceedings of the IEEE, 103(1), 14-76.
[2] Sahoo, K. S., Mohanty, S., Tiwary, M., Mishra, B. K., & Sahoo, B. (2016, August). A Comprehensive Tutorial on Software Defined Network: The Driving Force for the Future Internet Technology. In Proceedings of the International Conference on Advances in Information Communication Technology & Computing (p. 114). ACM.
[3] Sezer, S., Scott-Hayward, S., Chouhan, P. K., Fraser, B., Lake, D., Finnegan, J., ... & Rao, N. (2013). Are we ready for SDN? Implementation challenges for software-defined networks. IEEE Communications Magazine, 51(7), 36-43.
[4] Sahoo, K. S., & Sahoo, B. (2016). SDN Architecture on Fog Devices for Realtime Traffic Management: A Case Study.
[5] Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., & Shenker, S. (2008). NOX: towards an operating system for networks. ACM SIGCOMM Computer Communication Review, 38(3), 105-110.
[6] Wallner, R., & Cannistra, R. (2013). An SDN approach: quality of service using big switch's floodlight open-source controller. Proceedings of the Asia-Pacific Advanced Network, 35, 14-19.
[7] Erickson, D. (2013, August). The beacon openflow controller. In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking (pp. 13-18). ACM.
[8] Koponen, T., Casado, M., Gude, N., Stribling, J., Poutievski, L., Zhu, M., ... & Shenker, S. (2010, October). Onix: A Distributed Control Platform for Large-scale Production Networks. In OSDI (Vol. 10, pp. 1-6).
[9] Song, H., Gong, J., Chen, H., & Dustzadeh, J. (2014). Unified POF programming for diversified SDN data plane. arXiv preprint arXiv:1405.0060.
[10] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., ... & Turner, J. (2008). OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review, 38(2), 69-74.
[11] Sahoo, K. S., Sahoo, B., & Panda, A. (2015, December). A secured SDN framework for IoT. In 2015 International Conference on Man and Machine Interfacing (MAMI) (pp. 1-4). IEEE.
[12] Kim, H., & Feamster, N. (2013). Improving network management with software defined networking. IEEE Communications Magazine, 51(2), 114-119.
[13] Tiwary, M., Sahoo, K. S., Sahoo, B., & Misra, R. (2016). CPS: a dynamic and distributed pricing policy in cyber foraging systems for fixed state cloudlets. Computing, 1-17.
[14] Sahoo, A. K., Sahoo, K. S., & Tiwary, M. (2014, October). Signature based malware detection for unstructured data in Hadoop. In Advances in Electronics, Computers and Communications (ICAECC), 2014 International Conference on (pp. 1-6). IEEE.
[15] Sahoo, S., Nawaz, S., Mishra, S. K., & Sahoo, B. (2015, December). Execution of real time task on cloud environment. In 2015 Annual IEEE India Conference (INDICON) (pp. 1-5). IEEE.
[16] Panigrahi, C. R., Tiwary, M., Pati, B., & Das, H. (2016). Big Data and Cyber Foraging: Future Scope and Challenges. In Techniques and Environments for Big Data Analysis (pp. 75-100). Springer International Publishing.
[17] Han, B., Gopalakrishnan, V., Ji, L., & Lee, S. (2015). Network function virtualization: Challenges and opportunities for innovations. IEEE Communications Magazine, 53(2), 90-97.
[18] Wallner, R., & Cannistra, R. (2013). An SDN approach: quality of service using big switch's floodlight open-source controller. Proceedings of the Asia-Pacific Advanced Network, 35, 14-19.
[19] Pfaff, B., Pettit, J., Koponen, T., Jackson, E., Zhou, A., Rajahalme, J., ... & Amidon, K. (2015). The design and implementation of open vswitch. In 12th USENIX symposium on networked systems design and implementation (NSDI 15) (pp. 117-130).
[20] Addya, S. K., Sahoo, B., & Turuk, A. K. (2015). Virtual Machine Placement Strategy for Cloud Data Center.

[21] Mishra, S. K., Deswal, R., Sahoo, S., & Sahoo, B. (2015, December). Improving energy consumption in cloud. In 2015 Annual IEEE India Conference (INDICON) (pp. 1-6). IEEE.
[22] Puthal, D., Sahoo, B. P. S., Mishra, S., & Swain, S. (2015, January). Cloud computing features, issues, and challenges: a big picture. In Computational Intelligence and Networks (CINE), 2015 International Conference on (pp. 116-123). IEEE.
[23] Raj,K, Tiwary,M, Singh, A.,Sahoo,K.S.,& Sahoo,B.(2016,December). Improving Quality of Services During Device Migration in Software Defined Network. In Proceedings of ICACIE 2016. Springer.
[24] Sonkoly, B., Gulyas, A., Nemeth, F., Czentye, J., Kurucz, K., Novak, B., & Vaszkun, G. (2012, October). On qos support to ofelia and openflow. In *2012 European Workshop on Software Defined Networking* (pp. 109-113). IEEE.
[25] Egilmez, H. E., Dane, S. T., Gorkemli, B., & Tekalp, A. M. (2012). Openqos: Openflow controller design and test network for multimedia delivery with quality of service. *Proc NEM Summit, Implementing Future Media Internet Towards New Horiz*, 22-27.
[26] Jarraya, Y., Madi, T., & Debbabi, M. (2014). A survey and a layered taxonomy of software-defined networking. *IEEE Communications Surveys & Tutorials*, *16*(4), 1955-1980.
[27] Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, *103*(1), 14-76.
[28] Xie, J., Guo, D., Hu, Z., Qu, T., & Lv, P. (2015). Control plane of software defined networks: A survey. *Computer Communications*, *67*, 1-10.
[29] Jammal, M., Singh, T., Shami, A., Asal, R., & Li, Y. (2014). Software defined networking: State of the art and research challenges. *Computer Networks*, *72*, 74-98.
[30] Misra, R., Panda, B., & Tiwary, M. (2016, March). Big data and ICT applications: A study. In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies* (p. 41). ACM.

Author(s) Profile

Kshira Sagar Sahoo is a Ph.D. scholar in Computer Science and Engineering at National Institute of Technology, Rourkela, India. He received his M.Tech degree in Information and Communication Technology from IIT, Kharagpur, India in 2014. His research interests include Software Defined Network, network virtualization, Big Data Analytics. He is a member of IEEE computer society.

Sambit Kumar Mishra is pursuing a Ph.D. in Department of Computer Science & Engineering at National Institute of Technology, Rourkela, India. His area of research is Cloud Computing, Parallel and Distributed computing System, Wireless Sensor Networks. He obtained his M.Tech. and M.Sc. in Computer Science from Utkal University, India. He is a member of IEEE computer society.

Sampa Sahoo is pursuing a Ph.D. in Department of Computer Science & Engineering at National Institute of Technology, Rourkela, India. Her area of research is Cloud Computing, Parallel and Distributed computing System. She obtained her M.Tech. and B.Tech. in Computer Science & Engg. from Berhampur University, India and Biju Patnaik University of Technology (BPUT), India respectively. She has 7 years of teaching experience. She is a member of IEEE computer society.

Dr. Bibhudatta Sahoo obtained his M. Tech. and Ph.D. degree in Computer Science & Engineering from NIT, Rourkela. He has 24years of Teaching Experience in undergraduate and graduate level in the field of computer Science & Engineering. He is presently Assistant Professor in the Department of Computer Science & Engineering, NIT Rourkela, INDIA. His technical interests include Data Structures & Algorithm Design, Parallel & Distributed Systems, Networks, Computational Machines, Algorithms for VLSI Design, Performance evaluation methods and modeling techniques Distributed computing system, Networking algorithms, and Web engineering. He is a member of IEEE & ACM.