



# Software Defined Networking (SDN) for Universal Access

Adama Nantoume<sup>(✉)</sup>, Benjamin Kone, Ahmed Dooguy Kora,  
and Boudal Niang

Ecole Supérieure Multinationale des Télécommunications,  
Dakar BP10000, Senegal  
{adama.nantoume, ahmed.kora, boudal.niang}@esmt.sn,  
benjikone@yahoo.fr

**Abstract.** Ensuring Universal Access/Universal service to the populations of developing countries is up to now a big problem which can be explained by the fact that the telecommunication operators estimate certain areas unprofitable. The Universal access to the Technologies of Information and Communication being a non-discriminatory right for any citizen wherever he lives, different approaches are implemented to guarantee it. One of these approaches is to recourse on cheap equipment associated with innovating technologies. The aim of this article is to be able to study to what extent the Software Defined Networking could be a viable solution for the localities interested in Universal access. To reach this goal, we have been lead to study a typical Voice over IP traditional architecture and the architecture tooled with the Software Defined Networking technology.

When implemented the Software Defined Networking technology is supposed to guarantee a good quality of service. In our contribution we have set up a Voice over IP environment with Asterisk server as equipment of the network core, and affordable equipment such as the WIFI access points in the element entitled, «Collecting subscribers». The Quality of Service being our preoccupation, the comparison of all our results shows that the architectures with Software Defined Networking offer a better quality of services.

**Keywords:** OpenFlow · Quality of service (QoS)  
Software Defined Networking (SDN) · Universal access/Universal service  
Voice over IP (VoIP) · WIFI

## 1 Introduction

Universal access is a problem common to all the countries. The level of progression varies from one country to another. The poor profitability of certain areas arouses an a priori reluctance of operators who refuse to deploy their equipment in certain areas of the territory.

Being concerned only in gaining money, they actively look for ways to reduce their *capital expenditure* (CAPEX) and their *operational expenditure* (OPEX).

One of the most used approaches is the recourse on less costly equipment associated to a set of services matching the needs of the targeted localities.

The concern in this article is to be able to determine to what extent the Software Defined Networking (SDN) could be a viable solution for the areas interested in Universal access. To achieve this goal, a comparative study will be conducted between simple network architectures using SDN with the traditional architectures. The SDN architecture is supposed to guarantee an acceptable quality of service. Our contribution was partly about configuring different architectures with affordable equipment such as WIFI and a network core based on Asterisk. The quality of the service has been our preoccupation. We have chosen the most known cases of use which are the voice and the data to measure their Key performance Indicators (KPI) in a traditional environment, and in another one containing an SDN structure with OpenFlow. Both results will then be compared.

To do this work properly, we will proceed as follows:

- in the first part, we will give a general presentation of the SDN,
- in the second part, we will give a description of the OpenFlow protocol,
- in the third part, we will show our approaches
- in the fourth part, we will display our different results and will analyze them before concluding.

## 2 Generalities on the SDN

The SDN generality is a new paradigm in the network field. It is a technology in full growth. Due to its young age, it can take several meanings, according to the field. The most common meaning is the one given by Open Networking Foundation (ONF). ONF is a foundation which has much worked at the implementation of this technology. According to it, the SDN is defined as being an architecture in which the control plan and the data plan are uncoupled: the intelligence of network state is logically centralized, and the infrastructure abstracted from the applications [1].

The main idea of the SDN, as in the definition, consists in separating the function of transmission from the network, carried out by the control plan. Routers and switches are confined on the functions of packets switching. All the network services are implemented in only one controller common to several equipments. This separation allows to make the network much more flexible in its management, creates an ideal framework for the innovation as well as the progression of the network [2]. The SDN architecture presents three layers which are [3]:

- The **Infrastructure layer** is composed of physical and virtual peripherals of network. It is the lowest layer of the structure; it is composed of all the knots of the network, which communicate with the control layer through the OpenFlow Protocol,
- The **Control layer** centralizes the intelligence of the network. Composed of *Application Programming Interfaces* (APIs) in charge of ensuring the communication between the control layer and the upper and lower layers, it has a global view of the network. It registers the peripherals (users and network), the details of interconnection between them and administers a data base of the flows. These

pieces of information allow it to instruct the peripherals and implement rules (routing, sharing of responsibilities, etc.).

- The **Application layer** gathers the service networks, the orchestration platform and the business application. The application layer is in charge of the applications necessary to the customers and their requirements in terms of network, storing, calculation, safety and administration. In other words, it is the business layer. It is the layer that makes the two other layers function. At this level, the applications communicate with the controller through the APIs.

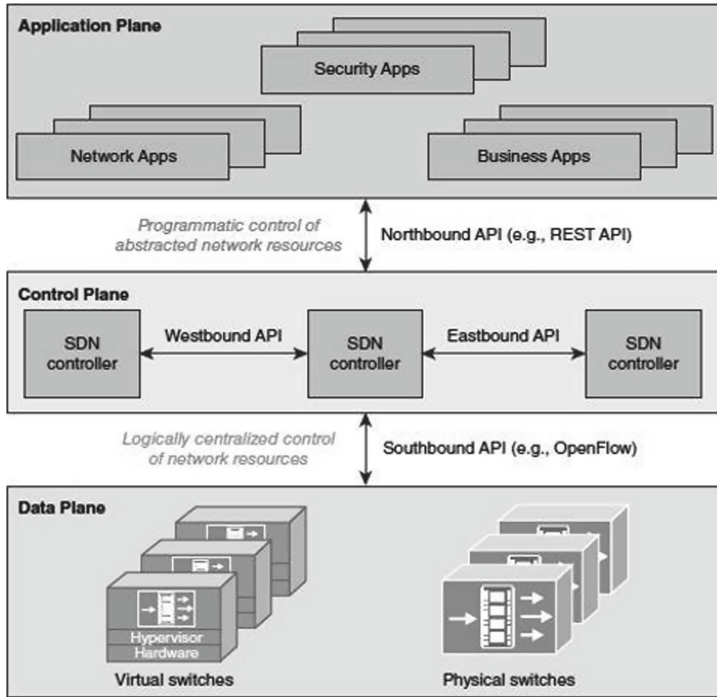


Fig. 1. SDN architecture [4]

## 2.1 The Interfaces

The south interface (southbound) provides the logical connection (signaling) necessary between the SDN controller and the commutators of the data plan. (see Fig. 1)

The north interface (northbound interface) provides the communication between the application level and the controller. Its aim is to define the needs of the application and to pass the commands to orchestrate the network with a good quality of service, appropriate safety and the required care.

### 2.2 The Abstractions

Scott Shenker, member of the board of the Open Networking Foundation and OpenFlow researcher, shows that the SDN can be defined by three fundamental abstractions [SHEN 1]: The forwarding, distribution and specification [4].

We note that one of the advantages of the decentralization of the controller is that the whole administration of the network is centralized and configurable. The modifications can dynamically be operated according to the needs. The SDN technology also allows flexibility to the network eliminates dependence on certain suppliers and allows a third organization to develop innovating application networks [5]. In addition to non-dependence to a supplier, the SDN enables the operator to have at their disposal very affordable equipment according to their need [6].

### 3 The OpenFlow Protocol

The OpenFlow protocol defines the communication between a controller and an OpenFlow commutator through a secured channel of the Secure Sockets Layer/Transport Layer Secure (SSL/TLS) kind to authenticate the two extremities. On the point of view of safety, this allows to minimize the risks of attacks during a communication [7]. OpenFlow, as such, is composed of a set of protocols and of an application programming interface (API). The protocols are divided in two parts, as shown on Fig. 2 [8].

- The OpenFlow protocol is also called filarial protocol. This defines a message infrastructure which enables the controller to include, to update and to cancel entries in the flow table [6],
- The OpenFlow logical Switch is a peripheral which defines the administration and configuration protocol via the abstraction layer called OpenFlow Logical Switch. It allows a wide access by assigning physical commutation ports to a specific controller (Fig. 3).

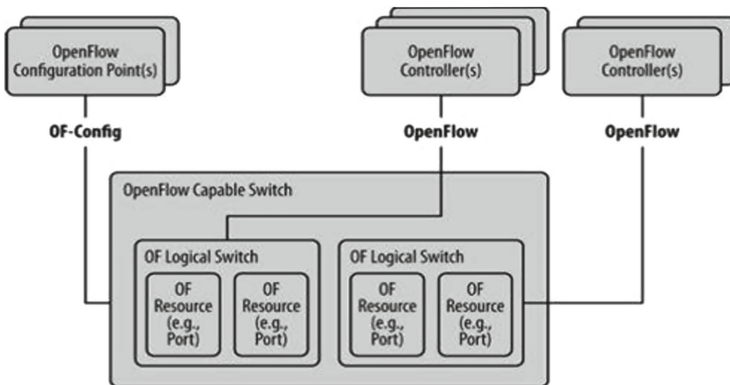


Fig. 2. Relationship between OF CONFIG and OpenFlow protocol [8]

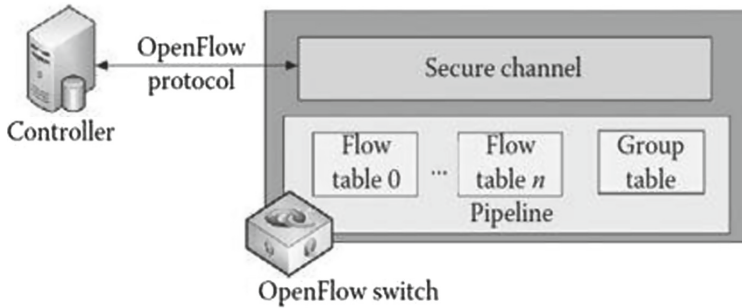


Fig. 3. OpenFlow general architecture

### 3.1 The OpenFlow Ports

With the OpenFlow protocol the ports are very important because it is through them that the packs get in and out. The connection of two OpenFlow switches is done with the ports. There are three types of ports in the OpenFlow protocol which are the physical ports, the logical ports and the reserved ports [10]. These ports can, at the same time be used as entry ports and exit ports for a pack.

- The physical ports for a pack are ports that can be seen on network equipment (e.g. Ethernet port),
- The logical ports are not linked to a physical port on the commutator. However the logical ports can be configured to correspond to a physical port on the commutator. When the packs are treated by the OpenFlow commutator, the physical and logical ports are treated the same way.

The reserved ports are specific ports used to engage a specific action. This action is started by sending a pack to a reserved port. An OpenFlow commutator is conceived to take care of five types of reserved ports. There are three other types of optional reserved ports which can be in charge of the commutator.

### 3.2 Flow Table

An OpenFlow commutator is composed of one or more flow tables and of an OpenFlow group table. Each table is composed of a set of flow entries. A flow entry is composed of a set of match fields, counter and actions (or actions). A flow entry is composed of:

- **Match fields:** Correspondence fields which define the pattern of pack flows through the instantiation of the heading fields, from the Ethernet layer to the transport layer,
- **Counters:** counters on the packets,
- **Actions:** actions to apply on the packets which correspond to the flow entry (Fig. 4).



Fig. 4. Structure of a flow entry

### 3.3 OpenFlow Message

In essence, the protocol is composed of a set of messages which are sent from the controller to the commutator, and of a corresponding set of messages which are sent on the opposite direction. There are three types of those messages [10].

- Messages from the controller to the switch, messages sent from the switch to the controller to administer or inspect its state. It also administers the configuration of the switches, the configuration of the roles, the configuration of asynchronous messages and many others [10].
- The asynchronous messages are sent from the switch to the controller without the switch being solicited by the controller. For example, when the switch receives a new packet, it will send a packet-in message to the controller in order to ask what section must be applied on the packet.
- The symmetric messages are generated in both directions (either by the controller or the switch) without the switch being solicited by the controller. For example, Hello (used at the same time by the switch and the controller), Echo (used at the same time by the switch and the controller) and Error (used in the switch) are messages which are classified in the symmetric messages (Fig. 5).

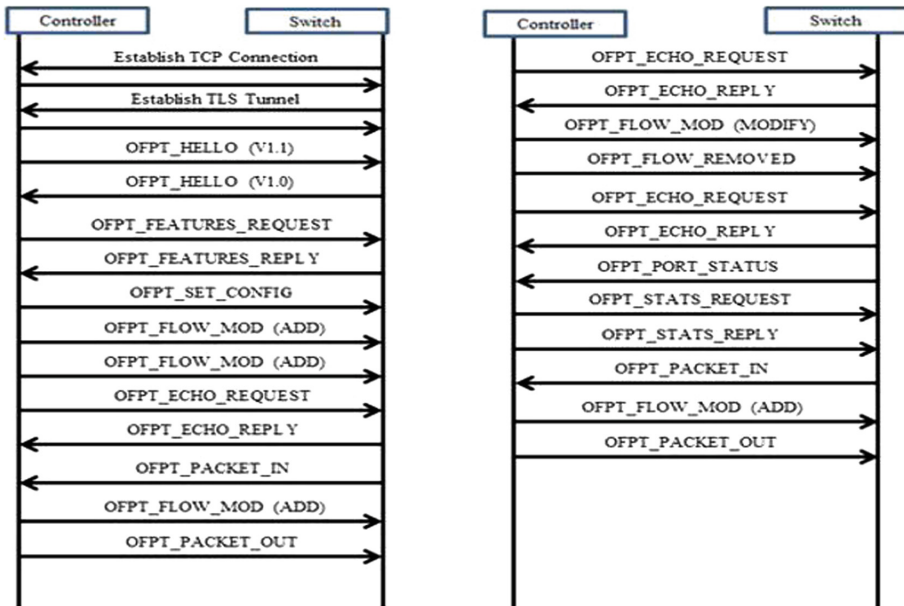


Fig. 5. Communication between OpenFlow switch and controller

## 4 SDN Implementation

We have realized SDN implementation in two steps in our approach.

### 4.1 First Step

The first one consisted in integrating the OpenFlow protocol on the different access points (the TP-link WR 1043nd). This integration has required the use of the Pantou method and OpenWrt.

In fact Pantou transforms an access point or a wireless router into compatible OpenFlow equipment [12]. Thus, fundamentally with Pantou, OpenFlow functions on OpenWrt as an application. OpenWrt is an Open source extensible exploitation system designed for the router, which is entirely customizable for the needs of the users and developers.

### 4.2 Second Step

A second process was about choosing an SDN router. In fact there are many of them on the market (license for sale or free license) [13], among which OpenDaylight, POX, NOX, RYU....

We opted for the RYU controller which provides software components, with well-defined API, which allow the developers to create easily new management and network control applications [14]. This approach by components helps the companies personalize the deployments to meet their specific needs. Developers can rapidly and easily modify the existing components or implement their own to make sure that the underlying network can meet the changing requirements of their applications. The programming with RYU is done in Python language, which we used to control the access to the different access points.

In this first architecture, we have the classical case of an Asterisk environment. To accede to the server, we have used two access points (TP link wr1043nd).

The customer is placed on either side of the access points during the different tests. The links between different entities are ensured with the help of Ethernet cables.

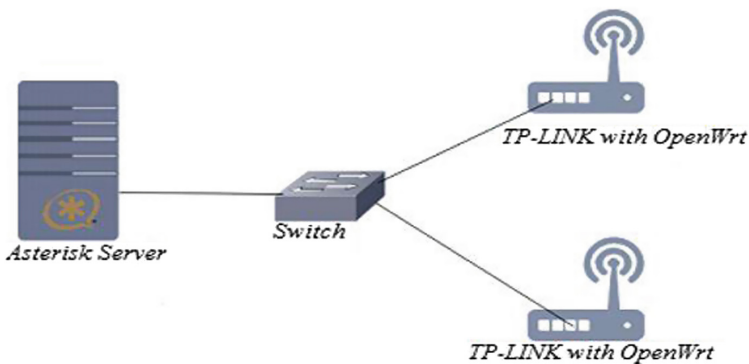


Fig. 6. Architecture without integration of SDN

We note that we are in a local environment, that means that server and access points are in a same network.

This case (Fig. 6) is without integration of the SDN. Like all access points, the screening policies, of fire guard and even the routings are configured at the level of the graphic interfaces of the access point. We can then say that the control part is at the level of the access points.

In the second case (Fig. 7) underneath, we have proceeded by the modification of the functionalities of the TP LINK router in order to integrate those of the SDN. First, we have changed the firmware of the two TP LINK access points into OpenWrt, and then we have installed the OpenFlow protocol as an application.

On the contrary of the first case, here the control part is centralized in a RUY controller. On the second, are programed the different access policies, the routing of the packets, etc. Python is the programming language which is used to implement these policies.

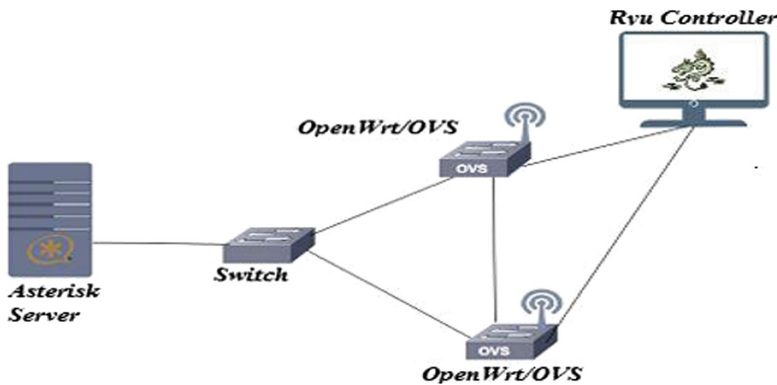


Fig. 7. Architecture with integration of SDN

## 5 Test and Analysis of Results

In comparing two architectures, we have done a series of tests in each configuration. We will collect the out coming results which will then be compared.

The tests are about the audio calls as well as the video calls done on the WIFI.

For the audio calls, we have used the StarTrinity Sip which is a software allowing call generation in an automatic way. On each call, it gives precise information on the QoS parameters, namely the number of packets, the lost packets, etc.

We have executed 50 audio calls on a row lasting one and half a minute (1 mn 30).

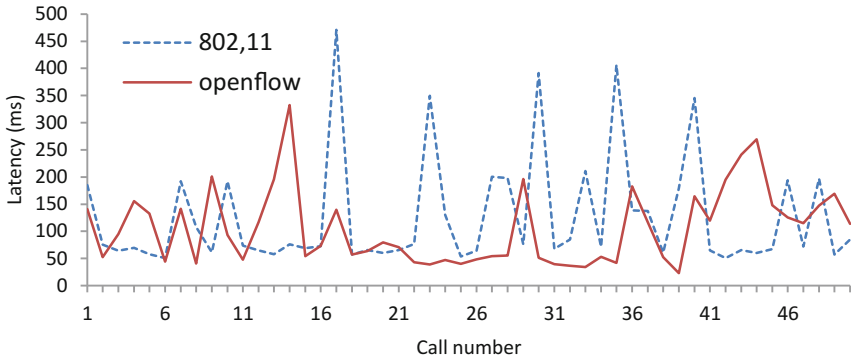
For the video calls, we have used the sniffing tool Wireshare to collect the different QoS parameters.

We have done thirty (30) video calls lasting 2 min.

In all cases graphs will be presented to highlight the variation of the latency on the customer's side.



The variation of the latency (simple WIFI and OpenFlow) of the fifty audio calls is presented in the graph below.



**Fig. 8.** Audio calls latency

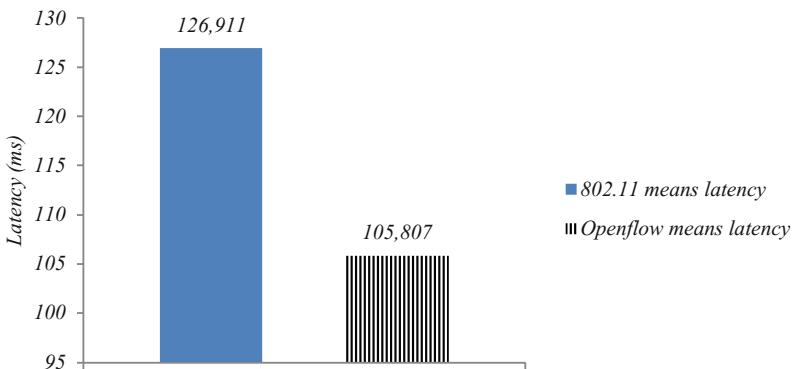
On the Fig. 8 we notice a neat difference between the variation of the latency of the WIFI and OpenFlow, which does not present any correlation in terms of nature of the curves however. On the other hand, we notice that at the level of amplitudes, the 802.11 presents higher pikes widely exceeding the maximal amplitudes of the OpenFlow.

Synthesis:

In terms of voice quality (MOS) we have noticed that the two technologies (802.11 and OpenFlow) offer a good quality of perception to human ear.

No loss of packet is registered during the calls.

To appreciate the difference at the level of the latencies an average of 802.11 and OpenFlow latency seems to us necessary. The curve below represents the average of both latencies (Fig. 9).



**Fig. 9.** Audio calls latency average

The curve enables us to notice a very good improvement of the latency with the integration of the OpenFlow protocol, although the average values of latency, according to ITU are considered very well (<150 ms).

The same type of measurements by making calls is used for the video calls. The variation of the latency of these different video calls is also represented on the graph underneath (Fig. 10):

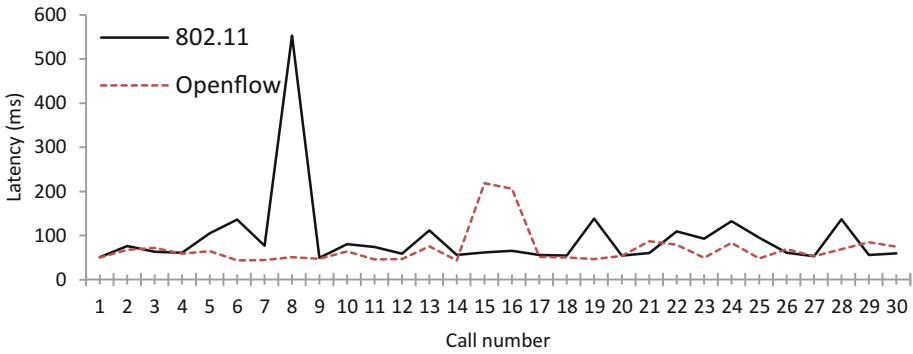


Fig. 10. Video call latency

We notice that at the level of the video calls, the latency curve 802.11 presents values higher than that of the OpenFlow on a great number of points (number of calls).

It is to be noted that the sequential of the values of the latency are due to the number call simulations operated on the server.

**Synthesis**

Just like the audio calls in terms of voice quality (MOS) we have noticed that the two technologies (802.11 and OpenFlow) offer a good quality of perception to human ear.

No loss of packet is registered during the calls.

Both technologies present a good value of latency; however the curve also enables us to notice a good improvement of the latency with the integration of the OpenFlow integration (Fig. 11).

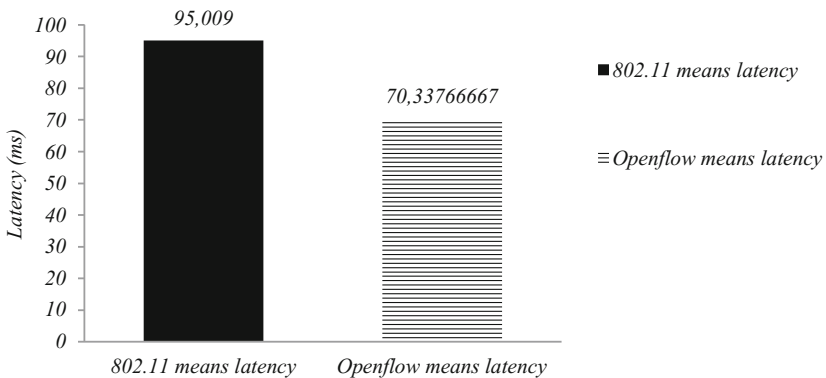


Fig. 11. Video call means latency

## General Synthesis

We make all measurements in the conditions. The obtained results show that traditional VoIP network and network with SDN implementation offer different quality of service. By comparing results, we notice that with either video or audio calls, the integration of the SDN causes an increase of the network performance (reduction of the latency).

## 6 Conclusion

At the end of our work about the implementation of SDN solutions in order to facilitate, if not guarantee Universal Access, the different solutions that we have implemented were based on the use of very affordable equipment as well as on the advantages that the SDN technology offers, namely the use of Cloud. What motivated the experimentation of such solutions is to enable the handicapped areas (the rural areas) to benefit from the information and communication technologies (ICT) services. In fact, operators are very reluctant when it comes to serving those areas which are considered unprofitable (low population density) because the CAPEX and OPEX equipment cost is relatively high.

In one word, we can say that the SDN brings about not only an improvement in terms of quality of service (QoS) in the network, but also a decrease in CAPEX and OPEX. Thus, it is an ideal solution for the areas which are ready for Universal access and Global Service.

## References

1. <https://www.opennetworking.org/sdn-definition/>. Accessed Oct 2017
2. Kreutz, D., Ramos, F.M.V., Verissimo, P.E., Rothenberg, C.E., Azodolmolky, S., Uhlig, S.: Software-defined networking: a comprehensive survey. <https://arxiv.org/pdf/1406.0440>. Accessed Aug 2017
3. <https://www.sdxcentral.com/sdn/definitions/inside-sdn-architecture/>. Accessed Oct 2017
4. Foundations of Modern Networking SDN, NFV, QoE, IoT, and Cloud de William Stallings. Addison Wesley 2015-10-29
5. Nugroho, A.S., Safitri, Y.D., Setyawan, T.A.: Comparison analysis of software defined networking and OSPF protocol using virtual media. In: 2017 IEEE International Conference on Communication, Networks and Satellite (Commnetsat) (2017)
6. Mahesh, A., Chandrasekaran, A., ArunKumar, R., SivaKumar, K., Vigneshwaran, N.: Cloud based firewall on OpenFlow SDN network. <http://ieeexplore.ieee.org/document/8186699/>. Accessed Dec 2017
7. Pujolle, G.: Software Networks: Virtualization, SDN, 5G and Security, p. 84. Wiley-ISTE, Hoboken (2015)
8. Nadeau, T.D., Gray, K.: SDN: Software Defined Networkings, p. 237 (2013). [books.google.com](https://books.google.com)
9. Morreale, P.A., Anderson, J.M.: Software Defined Networking Design and Deployment, p. 110. CRC Press, Boca Raton (2015)
10. Hu, F.: Network Innovation through OpenFlow and SDN Principles and Design, p. 96. CRC Press, Boca Raton (2014)

11. International Journal of Advance Research in Computer Science and Management Studies Research Article/Paper/Case Study. [www.ijarcsms.com](http://www.ijarcsms.com)
12. <https://www.sdxcentral.com/projects/pantou-OpenWrt/>. Accessed Nov 2017
13. Doherty, J.: SDN and NFV Simplified A visual Guide to Understanding Software Defined Networking and Network Function Virtualization, pp. 462–466. <http://ptgmedia.pearsoncmg.com/images/9780134306407/samplepages/9780134306407.pdf>
14. ryu Documentation Release 4.21 p 3. <https://media.readthedocs.org/pdf/ryu/stable/ryu.pdf>