A comparison of Traditional Network and Software-defined Network schemes using OpenFlow protocol.

ANDRES FRANCO-ALMAZAN,
NICOLE FERNANDEZ-SORIANO,
SERGIO VIDAL-BELTRÁN
Escuela Superior de Ingeniería Mecánica y Eléctrica – Zacatenco,
Instituto Politécnico Nacional,
Ciudad de México, MEXICO
svidalb@.ipn.mx

Abstract: - Today, we are witnessing an increase in the number of services and applications demanded by users and it presents a challenge for traditional networks in terms of implementation, costs and operation. Software-defined networks emerge as an agile, flexible and programmable architecture, which places it as a solution in the implementation of network applications and services. This paper presents a comparison between the performance of traditional networks and Software-defined networks applying tests to evaluate the parameters defined by the RFC 2544 methodology, in order to obtain a complete characterization of each scheme using a proposed topology applied to each technology, after that we analyze the performance of each one and examining its advantages and disadvantages. Currently, Software-defined networks and virtualization are a subject of many researches, as they are considered essential to counteract the increase in complexity and costs of managing and operating traditional networks.

Keywords: - software-defined network, virtualization, OpenFlow, Opendaylight, Mininet

1. Introduction

Internet has grown in recent years, data transmitted by the users increase considerably that provoke than traditional IP networks needs to be more complex (to defined policies inside the network) and hard to manage (to reconfigure each individual network device if network has a fault or uses a vendor-specific commands). The traditional IP networks needs to respond of the current requirements, this include the support in the increase of users that since some years increase for the expansion of the digital world.

Software-defined Networks emerged as a new paradigm that pretend to change the vision of the traditional IP Networks for a better network resources management. Current network devices have a logical control and data planes together and the software-defined networks has another approach about the traditional architecture, where reduce implementation and operating costs and distributing the traffic of each element in the network topology to do it more efficient. Software-defined network responds to the need for a new generation of communications that allows adapting to new requirements such as latency less than 1ms or the increase in the throughput in the network [1].

1.1 Related Work

Several papers of software-defined networks and its architecture had been written. In [2] the authors describe a fundaments, concepts, future related works and research opportunities and challenges of software-defined networks, this allow to identify definitions and applications for this topology. For [3] authors analyze the SDN architecture and how the planes can communicate each other, that include OpenFlow protocol to communicate the controller with the data plane. Also, in [4] the paper describes each plane of the SDN architecture and the element that exist in every plane like a variety of controllers, but in this case the paper works with Mininet Emulator to generate the SDN topologies. These papers define general concepts of software-defined networks and their main features, other papers like [5] works with specific software define network algorithm to control data traffic, this algorithm was developed in the control plane and allow to manage data flows, it evaluated the performance of SDN topologies.

In this paper, we present a comparison between software-defined networks and Traditional

Networks. Firstly, the concepts of software-defined network and their architecture are defined in section II.

Proposed topology is shown and described to evaluate each scheme in section III. Then, the performance benchmarks within RFC 2544 methodology and network traffic are also specified in section IV. Finally, the results and discussions are shown in Chapter V and the conclusions are reported in Chapter VI.

2. Software-defined Network

Software-defined Networks can provide a dynamic and flexible architecture this is possible because the principal idea of SDN is centralize control and separate network control plane (where the decisions of handling traffic are made) and the data network plane (that forward the traffic between the network topology) [6]. These planes are communicating by a protocol, also the software-defined network are programmable, that means that the network administrator configure flows and policies directly in the network topology by the management plane (where the administrator decide the configuration of each device) in this plane the users can configure the flows of every virtual switch, it depends on the topology created. Once created, the settings go through an entity of the control plane called controller that manage a table flow into the virtual switch, this table define a path that a flow takes from source to destination regardless of the network topology, and utilizes flow-based processing for forwarding packets with the decisions taken by the administrator. In the environment of software-defined networks include a term like a *virtualization* that its referred as process of creating a software-based, or virtual, representation of something, such as virtual applications, servers, storage and networks, network virtualization [7] allow to use the resources of a host system with the aim of installing guest systems that can create network topologies with help of different virtualization tools. This technology can provide a low cost of implementations, centralize control in an entity and open and flexible programmability. SDN makes it easier to create and introduce new abstractions in networking, simplifying network management and facilitating network evolution.

For the evaluation of software-defined networks, we use the architecture as shown in Fig. 1, this figure includes virtualization tools applied in each plane of the architecture SDN [8]

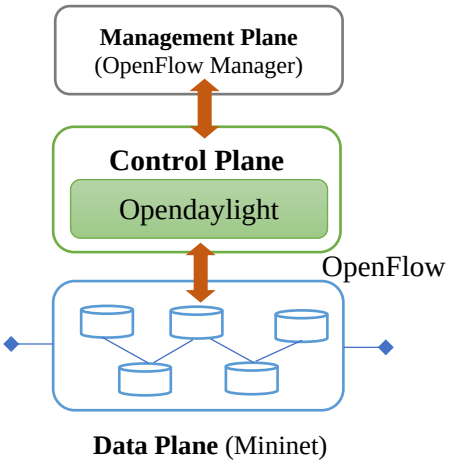

Fig. 1. SDN Architecture and used tools.

The main tools to develop SDN architecture are the following: the northbound plane (on top), use the OpenFlow manager application its developed by Cisco [9], this application allow to monitoring, configure and manipulate the elements in the network SDN, this include creation and delete of the flows table into the virtual switches. Other important tool is Opendaylight Controller located in the control plane and developed by Linux foundation, it's an initiative SDN controller module platform java-based and gives the control of the topology [10]. Between control plane and data plane exist a protocol called OpenFlow [11] that help to communicate the changes and configuration from controller to data plane devices using the TCP port 6633 for remote programming. Mininet is a program for generate and emulate the network devices in the data plane, this emulator is a project to adapt and develop software-defined networks this program allows to create OpenvSwitch devices (depending of the network topology) and connect with controller (in this case Opendaylight) [12]. It's important to mention that every element of the SDN architecture use the host system resources as a guest system, for this reason it needs an additional software that emulate all of them in a unique topology, this emulator is GNS3 that design, configure and execute different networks infrastructure like a SDN or a IP traditional networks with devices like a switches or routers, despite devices are limited by features and processing of the host system.

3. Topology Implementation

One topology has been designed to perform an evaluation of traditional networks and Software-defined network schemes, this topology was designed to be complex and it depends of the limitations of the host system where they are virtualized. Each element of SDN architecture will have the following features and resources as shown in Table 1:

Table 1. Resources and Configurations

ENTITY	CPU	RAM	OS
Mininet Opendaylight Cisco IOS	Intel Core i5-7200U @2.5GHz	2GB 1GB 512GB	Ubuntu 18.04
GNS3		1GB	Windows 10

3.1 Proposed Topology

This topology was designed with three layers where we find on the bottom the access layer, in middle distribution layer and core layer on the top of the topology, this layer has the aim of communicate lower layers in the same subnet, which is 192.168.10.0/24, this topology is shown in Fig. 2. In Fig. 3a shown SDN emulation, where use Mininet and its connected with the Controller Opendaylight.

In this case, Mininet create the layers of the proposed topology using OpenvSwitches and communicated to controller Opendaylight by OpenFlow protocol version 1.3 this include OpenFlow Manager application, SDN Network was designed in GNS3 included a Virtual Machines with run an SDN tools and applications on OS Ubuntu 18.04. Proposed topology in traditional networks was emulated by IOS Cisco with version 15.4, in this case have a 7 switches, each switch have 8 port- Gigabit Ethernet , that its connected emulating a layers of the topology and on the bottom of the topology has connected a hosts, two hosts are virtual machines with Ubuntu Server 18.04 configurated with Gigabit Ethernet ports as shown in Fig. 3b.

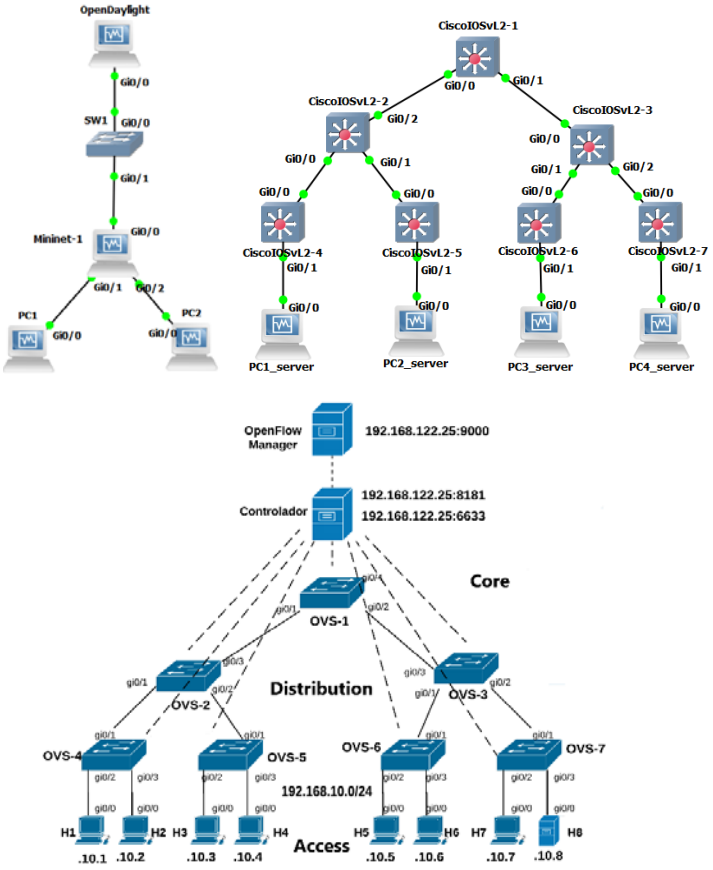

Fig. 2. Proposed Topology

a) *b)*

Fig. 3. a) SDN Emulation b) Traditional Network Emulation

4. Performance Parameters

For evaluation of both schemes, SDN and Traditional Networks, we use some parameters to evaluate the performance of each architecture, with these parameters we can make a comparison. The parameters are defined by RFC 2544 [13], it's a methodology used to describe the performance of network interconnect devices. This methodology takes relevant factors such as throughput, latency, frame loss and Jitter [14], also work with specific frame sizes for each test with variations of congestion load in the evaluated networks.

These tests were performed through a point-to-point communication at each end during 60 seconds and waiting 2 seconds for every test and repeating evaluation for 10 times. Each parameter is evaluated with a different frame size that is defined within the RFC 2544 methodology; these values are defined by

the transport technology used in this case will be Ethernet. These values are 64, 128,512, 1024 and 1518 bytes to be able to characterize the network performance. For the throughput, duration of the test should be at least 60 seconds and focuses on the maximum speed of the link and in the case of the latency parameter and jitter the test should be repeated at least 10 times and an average is obtained. In the case of the Frame Loss parameter, a certain number of frames are sent in a time interval depending on the frame size, in this paper a defined data value is sent and an average is also obtained,

4.1 Network Traffic

For evaluate the performance of proposed topology created with both scheme (SDN Network and Traditional Networks) we congest each network with different traffic data types, these data emulate a traffic inside the network. The traffic data is generated by a host as a client and it received by a host as a server within the network. The data traffic is described in the following Table:

Table 2. Network Traffic

NETWORK TRAFFIC	CONCEPT
Without Traffic	In this case. There is no additional traffic
HTTP Traffic	This kind of traffic is generated by request and responses HTTP client to a server connected to the network, this request gets HTTP traffic and congest the network. To generate a congestion in every network layer, several requests are generated within network
RTP Traffic	Traffic RTP use the Real-time Transport Protocol to transmit data such as audio and video over IP networks.

5. Results

5.1 Throughput

To measure this parameter, we send a load of around 1024 MB in frames of different size (64, 128, 256, 512, 1024, 1518 bytes) and this frames are sending through the network from host 1 (H1) to host 7 (H7) into the topology network, these sizes include the maximum and minimum frame sizes allowed by the Ethernet. First, evaluation is performed without the additional network traffic and then testing include additional network traffic, described in the previous section, the network traffic is generated from host 2 (H2) as a client to host 8 (H8) as a server. In the case of the SDN network the best performance is located when the network does not include additional traffic as shown in Fig. 5a, maintaining a value of up to 315.12 Mbps when only 50% of the load is used and decrease to 305.36 Mbps when the full load network is included using frames of 1518 bytes. When the full load is sent with different size of the frame to a smaller value, the throughput parameter decreases as the size of the frame is also smaller, this is because it is necessary to increase the number of frames to be able to transport the full load, which can saturate the network and lower the performance of the network. In the case of traditional networks this behavior is similar but the values of the throughput parameter are different compared to SDN networks, as shown in Fig. 4a obtaining a value of up to 26.03 Mbps when only 50% of the load is used and decrease to 24.47 Mbps when the full load is included using frames of 1518 bytes. In both cases the behavior is the same when additional traffic is added to the network caused by the saturation existing in the network. In this level of throughput, the services as streaming or VoIP presents delays and transmission issues.

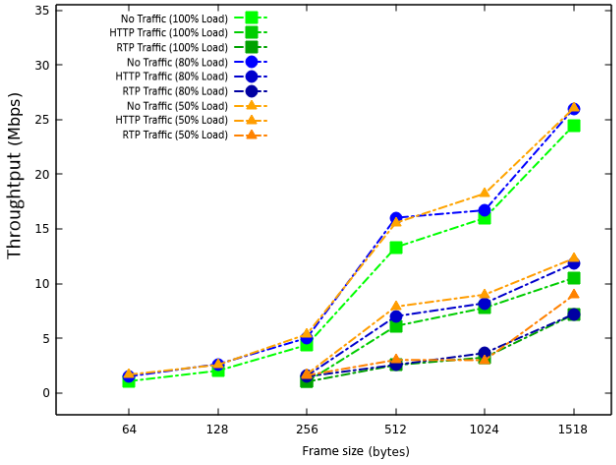

Fig. 4. Throughput in Traditional Networks

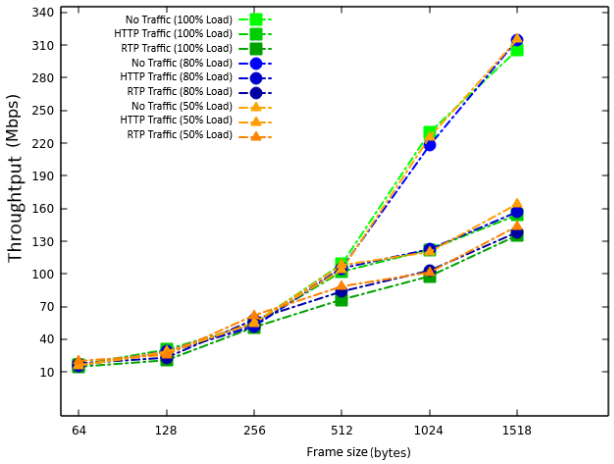

Fig. 5. Throughput in SDN Networks

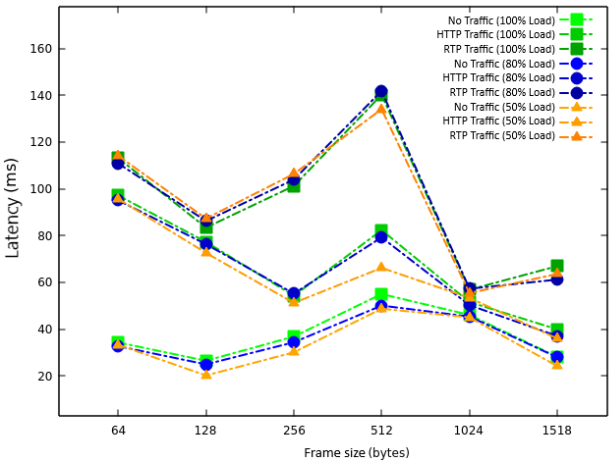

Fig. 6. Latency in Traditional Networks

5.2 Latency

To evaluated the latency in the network, a transmitter host send a packet with different conditions, in this case it sent 100 packets from host1 (H1) to host 7 (H7), also network traffic go through in to network from host 2 (H2) to host 7 (H7). In the case of the SDN network the best performance is located when the network does not include additional traffic but when additional traffic like RTP traffic is included, as shown in Fig. 7b, we have values up to 2.453 ms when only 50% of the load is used and increase to 4.711 ms when full load is included using frames of 1518 bytes, when frames of 128 bytes are used the values are better, obtaining values up to 1.564 ms when only 50% of the load is used and increase to 1.977 ms when full load is included. For traditional networks the values are different, as shown in Fig. 6b, obtaining values up to 214.02 ms when only 50% of the load is used and increase to 220.16 ms when full load is included using frames of 512 bytes, when frames of 128 bytes are used the values are better, obtaining values up to 97.24 ms when only 50% of the load is used and maintaining in 96.4 ms when full load is included. This is because when large frames are used for data transport, the latency increases by the size of the frame including network saturation that causes delays, in this case the frames of 128 bytes have a better performance than 64 bytes (minimum frame size) this result of the type of traffic. Latency is indispensable to implement Real-time services for this reason is important to evaluate it. If this parameter is higher, this generate packet loss problems.

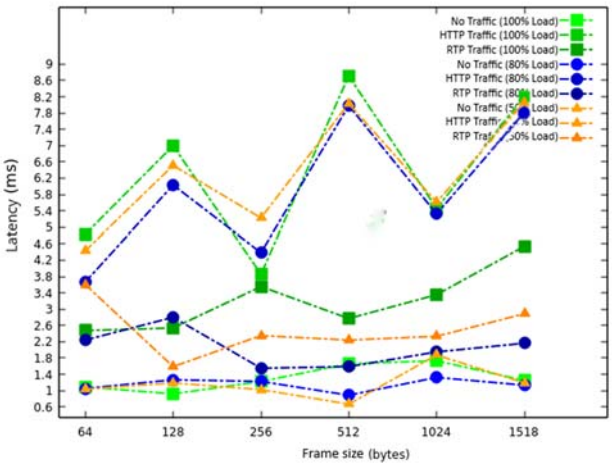

Fig. 7. Latency in SDN Networks

5.3 Frame Loss

The evaluation of frame loss parameter we send a limiting the number of bytes in frames of different size (64, 128, 256, 512, 1024, 1518 bytes) and this frames are sending through the network from host 1 (H1) to host 7 (H7) into the topology network, in SDN networks data transmitted are 200MB and traditional networks data transmitted are 3MB, because in this level both technologies present a stable connection. The frames are sent through the network to verify if its correct transmission or data loss with the information it generated a frame loss rate. Table IV and Table V shown a values of frame loss rate. The frame loss parameter is expressed in a percentage that indicates the frames lost within the network. In SDN networks, it can be observed that when HTTP traffic is included, the values are kept below 5% when the frame is smaller than 512 bytes, with higher values the frame loss is greater, reaching up to 42%. caused by network saturation with large size frames. The same behavior is observed in

traditional networks although the percentage of lost frames increases reaching up to 56% of loss frames, this high value causes delays in transmission, low throughput or incomplete information in the transmission.

Table 3. Frame Loss Rate in SDN Networks

Frame size (bytes)	**Frame sent (bytes)**	Without Traffic (%)	**HTTP Traffic** (%)	**RTP Traffic (%)**
64	2564103	0.175	1.09	35.87
128	1408451	0.2342	3.71	39.33
256	740741	0.992	5.39	37.19
512	380229	1.21	24.86	37.4
1024	192679	> 0.1	42.83	42.46
1518	132101	> 0.1	24.73	42.12

Table 4. Frame Loss Rate in Traditional Networks

Frame size (bytes)	**Frame sent (bytes)**	Without Traffic (%)	**HTTP Traffic** (%)	**RTP Traffic (%)**
128	24215	1.8	3.42	50.22
256	15812	1.9	2.04	51.32
512	9055	2.61	33.633	55.76
1024	2560	2.77	37.87	60.82
1518	2319	0.90	56.12	68.99

5.4 Jitter

To evaluate the parameter of Jitter, the test consists to send different frame size through the network like a latency parameter but in this case, variance in time delay are evaluated. Jitter parameter is indispensable to implement real-time services (RTP traffic) [15] like a Latency parameter, Jitter have a similar variations, as shown in Fig. 9c, with values up to 1.961 ms when only 50% of the load is used and increase to 3.150 ms when full load is included using frames of 1518 bytes, when frames of 128 bytes are used the values are better, obtaining variations up to 1.118 ms when only 50% of the load is used and increase to 1.243 ms when full load is included. For traditional networks the values are different, as shown in Fig. 8c, obtaining values up to 121.53 ms when only 50% of the load is used and increase to 131.45 ms when full load is included using frames of 256 bytes, when frames of 128 bytes are used the values are better, obtaining values up to 79.98 ms when only 50% of the load is used and maintaining in 78.06 ms when full load is included. These variations are caused both by the traffic within the network and by the delay provoked by the size of the frame. If the value of the Jitter parameter increases in the network, it will present delays in packet communication, this causes transmission issues.

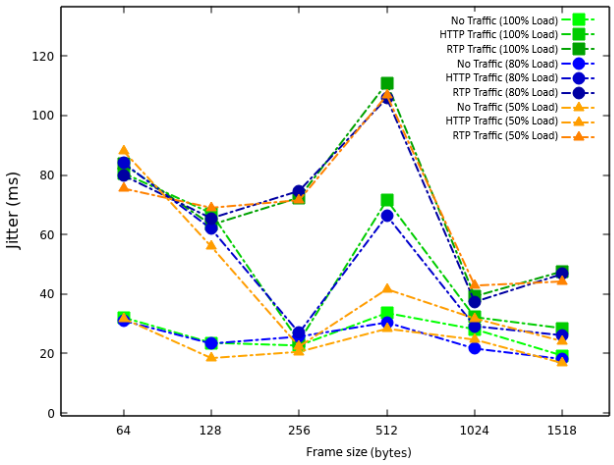

Fig. 8. Jitter in Traditional Networks.

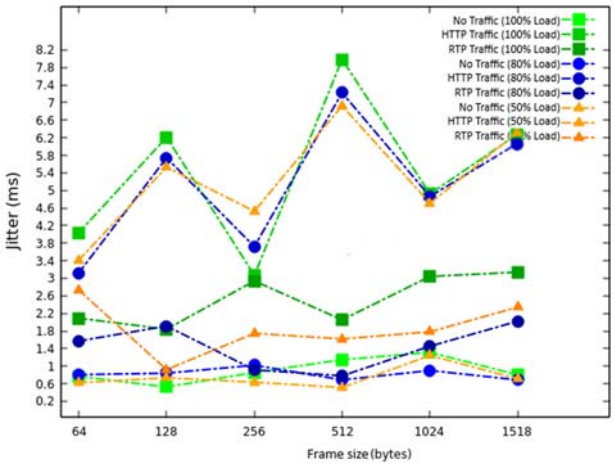

Fig. 9. Jitter in SDN Networks

6. Conclusion

Evaluation of the proposed topology have to aim to do a comparison of the Software-defined networks and Traditional Networks architecture with a methodology that pretend to be able to get a full characterization of the performance. This test congests the proposed topology with different frame size, different types of network traffic and with variations of load network

The test results obtained show a better performance in the SDN networks by decoupling the control plane it gets agility and flexibility as shown by throughput or latency parameters in addition to reliability because SDN have a smaller frame loss or in the case of Jitter parameter that affect the real-time services

over networks. The performance parameters used allow us to visualize the behavior of both schemes saturated to different types of traffic obtaining a behavior in a real environment.

References:

[1]. International Telecommunication Union, "Setting the scene for 5G: opportunities and challenges". [Online] Available: https://www.itu.int/net4/ITU-T/registration/Resolver/Index?handle_id=11.1002/pub/811d7a5f-en

[2] D. King, C. Rotsos, A. Aguado, N. Georgalas and V. Lopez, "The Software Defined Transport Network: Fundamentals, findings and futures," *2016 18th International Conference on Transparent Optical Networks (ICTON)*, Trento, 2016, pp. 1-4. [Online]. doi: 10.1109/ICTON.2016.7550669

[3] M. S. Olimjonovich, "Software Defined Networking: Management of network resources and data flow," *2016 International Conference on Information Science and Communications Technologies (ICISCT)*, Tashkent, 2016, pp. 1-3. [Online]. doi: 10.1109/ICISCT.2016.7777384

[4] V. Gupta, K. Kaur and S. Kaur, "Network programmability using software defined networking" *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, 2016, pp. 1170-1173.

[5] A. Srikanth, P. Varalakshmi, V. Somasundaram and P. Ravichandiran, "Congestion Control Mechanism in Software Defined Networking by Traffic Rerouting" *2018 Second International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, 2018, pp. 55-58. [Online]. doi: 10.1109/ICCMC.2018.8488041

[6] Citrix, "SDN 101: An introduction to software defined Networking". [Online] Available: https://www.citrix.com/content/dam/citrix/en_us/documents/oth/sdn-101-an-introduction-to-software-defined-networking-en.

[7] VMware, «Virtualization Overview» 2006. [Online]. Available: https://www.vmware.com/pdf/virtualization.pdf

[8] P. K. D. S. S. J. Haleplidis E, "Software-Defined Networking (SDN): Layers and Architecture Terminology". Patent RFC 7426, January 2015.

[9] Cisco, «OpenDaylight OpenFlow Manager (OFM) App» CiscoDevNet,[Online].Available: https://developer.cisco.com/codeexchange/github/repo/ CiscoDevNet/OpenDaylight-Openflow-App/

[10] The Linux Foundation Project, «Lithium: The new release,» Opendaylight Project, 2015. [Online]. Available: https://www.opendaylight.org/what-we-do/current-release/lithium.

[11] V. Padma, P. Yogesh, "*Proactive Failure Recovery in OpenFlow Based Software Defined Networks "in* 3rd International Conference on Signal Processing, Communication and Networking (ICSCN),2015, [Online]. doi: 10.1109/ICSCN.2015.7219846

[12] Mininet Team, «An Instant Virtual Network on your Laptop (or other PC)»Mininet,2015.[Online].Available: http://mininet.org/

[13] S. Bradner, J. McQuaid" Benchmarking Methodology for Network Interconnect Devices". Patent RFC 2544, March 1999

[14] S. Bradner, J. McQuaid," Benchmarking Terminology for Network Interconnection Devices". Patent RFC 1242, July 1991.

[15] Cisco, «Understanding Jitter in Packet Voice Networks(Cisco IOS Platforms)»[Online].Available:https://www.cisco.com/c/en/us/support/docs/voice/voice-quality/18902-jitter-packet-voice.html