Analyzing the Performance of Software Defined Networks vs Real Networks

Jose M. Jimenez, Oscar Romero, Albert Rego, Jaime Lloret
Universidad Politécnica de Valencia
Camino Vera s/n 46022, Valencia (Spain)
email: jojiher@dcom.upv.es, oromero@dcom.upv.es, alrema91@gmail.com, jlloret@dcom.upv.es

***Abstract*—Emulators and simulators provide an easy way to reduce hardware needs in experiments. Because of that, network researchers use applications that allow them to emulate or simulate networks, like Mininet in Software Defined Networks. It is desired to obtain very close results between the ones given in a virtual network and the ones obtained when the real network hardware is implemented in order to avoid using too much hardware in complex experiments without gathering unreal results. In this paper, we compare the experimental results obtained when a virtual network is generated by using Mininet versus a real implemented network. We have compared them varying the Maximum Transmission Unit (MTU) on Internet Protocol version 4 (IPv4) packets. Ethernet, Fiber Distributed Data Interface (FDDI), and Wireless Local Area Network 802.11 (WLAN 802.11) MTUs have been used in our experimental tests. We have worked with different link capabilities and generated traffic with different bandwidth.**

Keywords- SDN; OpenFlow; Mininet; MTU; virtualization; bandwidth; jitter.

I. INTRODUCTION

In the field of computer networks, the researches usually use programs that allow us to emulate or simulate networks. This is because, in most cases, we do not have the necessary devices needed to create complex networks, but we need to know if these programs are reliable [1]. There are emulators and simulators as Omnet++ [2], OPNET [3], NS-2 [4], NS-3 [5] Netsim [6], GNS3 [7], etc. that are frequently used to create computer networks.

Deployment of network is very quick in virtual environment, even if it is needed a large number of resources, which is always practically almost impossible to implement with real hardware. Problem solving or troubleshooting capability is still easier than real implementations. Note that a network researcher has to keep in mind that the results obtained from a virtual network should be similar from those obtained by the real hardware network. If there is a significant difference between results of virtual network and real network, then the research work should not be taken into consideration. As a network test bed gives almost the same results as the real implemented network, then it saves a large amount of time, complexity and a lot of resources.

In general, network devices perform the transport and the control function. But, configuring a great amount of devices and changing the configuration efficiently to work properly, it means a big challenge for networking professionals.

Today's, computer network world is able to offer a large amount of functionalities suited to the requirements of users. A new technology, named Software Defined Networking (SDN) [8] appears to increase the efficiency and reduce the cost of network configuration.

Figure 1 shows the components of SDN in a layered structure. The first layer consists of some frequently used tools of monitoring and depuration. The tool “Oftrace” is used for analyzing and parsing Openflow message from network dump. “Oftrace” provides a library which analyzes and parses the message from TCP dump or Wireshark [9]. Loops or cyclic path can cause critical problems in SDN. “Oflops” is a tool to catch the loop mechanism in the software defined networks. It mentions the data packets in the loop which are not able to leave the network [10]. “Openseer” is a CGI script which helps to plot that data effectively in SDN [11]. In Controllers Layer there are few controllers which are used in SDN. More often, controllers are called the Brain of Network which controls and manages the software defined network. Floodlight, Open Daylight, Beacon, Nox are among the frequently used controllers in SDN [12]. Flow Visor ensures that multiple isolated logical networks can share the same topology and hardware resources of a network. It places as a transparent proxy between OpenFlow switches and OpenFlow controllers. The isolated logical network is named slice of the network and flow visor is named slicing software in SDN [13]. In SDN environment, OpenFlow switches are used to forward the packets. OpenFlow switches are either a software program or a hardware device which is compatible with OpenFlow protocols. Some of the commercial switches are available in market like HP, Nec, Juniper, etc. [14]. Mininet is used to create realistic virtual network within seconds on a single machine that could be able to run real kernel, switch and application code [15].

There are few emulators and simulators which are frequently used to run and control the technology SDN from a single screen. Some of them are NS-3, Estinet 9.0 [16], OmNet ++, Mininet, etc.

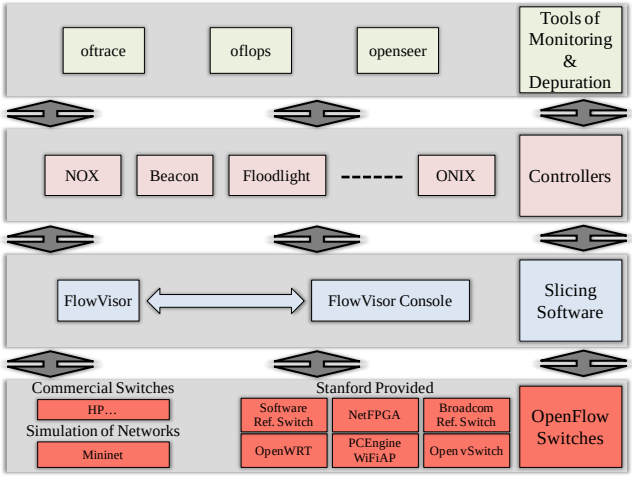

Figure 1.Key component of SDN in layered structure.

In this paper, we show the comparison among the obtained results from the virtual networks and from the real implemented networks. With the assessment of these results we are able to find the significant differences, which may be very useful for the researchers who all are performing their research work in Networking Industry. We have performed different experiments over Mininet and real implementation to have a good understanding of the network behavior in both scenarios. To do a detailed study, we must send data packets of different properties and compare the results. We used the data packets with different Maximum Transmission Unit (MTU) on IPv4. These sizes of packets are usual for Ethernet version 2, Ethernet with Logical Link Control (LLC), Point-to-Point Protocol over Ethernet (PPPoE), WLAN, Token Ring and FDDI.

This paper is an extended version of a conference paper published in [1].

The rest of the paper is structured as follows. In Section II, we discuss existing related works. In Section III, we introduced all resources that we used in our test bench. Measurement results and our discussion and analysis are shown in Section IV. Section V shows the conclusion and future works.

II. RELATED WORKS

In the past, a few researchers have accomplished their work in the area of SDN and investigated the performance of multimedia delivery over SDN. Furthermore, in the last years, emulators have been developed in order to provide an easy way to manage virtual networks and perform the research experiments. These emulators reduce the costs associated to the hardware needed to build the network. Inside the SDN research, the emulators have a great importance because of the great number of tests and the specific hardware that are necessary.

In the following section, we are going to discuss about some previous research work that helps us to get a deep understanding of SDN. Then, we will describe the previous researches in which emulators provide a useful way to test the experiments.

Recently, in our previous research article [17], we tried to evaluate the performance of multimedia streaming delivery over Mininet compared to real network implementation. We considered different properties of multimedia delivery, i.e., bandwidth, delay, jitter, and we found some significant differences over mininet and real test network. Kreutzet et al. [18] discussed the SDN, and analyzed the significance of SDN over traditional networking. Authors explained about the key components of SDN by using a bottom-up layered approach and focused on challenges, troubleshooting and debugging in SDN. Noghaniet et al. [19] introduced a framework based on SDN that could enable the network controller to deploy IP multicast between source and subscribers. The network controller was also able to control the distributed set of sources where multiple description coded (MDC) video content is available by using a simple northbound interface. Due to this SDN-based streaming multicast framework for medium and heavy workload, the Peak Signal-to-Noise Ratio (PSNR) of the received video is increasing considerably. Authors noticed that the received video, which had a very poor quality before, was having a significant increase in the quality of video now. Nam et al. [20] proposed a mechanism to solve the congestion problem and improve the video quality of experience (QoE). Authors tried to develop an SDN based application to improve the quality of video that can monitor conditions of network in real time streaming, and change routing paths dynamically by multi-protocol label switching (MPLS).

Egilmezet et al. [21] give a unique design of an Openflow controller for multimedia delivery over SDN with end to end Quality of Service (QoS) support. The authors tried to optimize routes of multimedia flows dynamically. After experiments over real test network, the authors found better results than HTTP based multi-bitrate adaptive streaming. They ensured that OpenQoS can guarantee the video delivery with little or no video artifacts experienced by the end-users. In another publication, Egilmezet et al. [22] gave new distributed control plane architectures for multimedia delivery over large-scale, multi-operator SDN. The extensions included in the design of architecture were: (a) to acquire network topology and the state information by topology aggregation and link summarization, (b) to propose an optimized framework for flowing based end to end over multi-domain networks, and (c) two distributed control plane designs by addressing the messaging between controllers for scalable and secure routing between two domains. By applying these extensions on layered video streaming, authors obtained a better quality of received video, reduced cost and memory overhead. This architecture was effectively scalable for large networks. Kassleret et al. [23] tried to negotiate the service and parameter for network communication between end users, and assign multimedia delivery paths in network according to prefixed service configuration. The idea behind this system was to centralize multi-user optimization of path assignments, which provides the better quality of experience by considering network topology, link capacities, delay and account service utility. Due to optimization, the system was able to use Openflow to set up forwarding paths in network.

In [24] Yang et al. have proposed a novel time-aware software defined networking (TaSDN) architecture for OpenFlow-based datacenter optical networks, by introducing a time-aware service scheduling (TaSS) strategy. The strategy can arrange and accommodate the applications with required QoS considering the time factor, and enhance the responsiveness to quickly provide for datacenter demand.

Dramitinoset et a1. [25] have discussed about different aspects of video delivery over next generation cellular networks, which includes the software defined networks and cloud computing. The authors have been focused on next generation cellular networks which employ SDN in core due to increased demands of video streaming commercially. In our paper we are trying to explore the performance of multimedia delivery over Software Defined Networks as compared to real test networks in terms of some important parameters.

Some of these researches test their experiments with emulators. As told before, in [17] we evaluated the performance of Mininet. Mininet is the most used emulator in SDN researches. In the paper "Using Mininet for Emulation and Prototyping Software-Defined Networks" [26], Oliveira et al. concluded that despite some limitations related with the fidelity of performance between the real network and the emulated one, Mininet has several positive aspects like the capacity of fast and simplified prototyping, the possibility of showing and sharing results, its applicability and low cost.

In addition, Wette et al. tried in [27] to create a large network emulation using Mininet. Their goal was to do an emulated network almost as large as e.g. data center networks, which are composed by thousands of nodes. They presented a framework called Maxinet, based on Mininet, able to emulate 3200 hosts in a cluster of only 12 physical machines, although they concluded that, even a larger network could be emulated with Maxinet using better hardware available than they used in the research.

This last research proves the power of emulators like Mininet, and the performance that can be obtained from Mininet shows why it has become the most used emulator, especially in the SDN field.

However, there are other emulators that are used in the literature in order to obtain the results of the research. In this section we are going to enumerate some cases where emulators are used to set up a virtual SDN. There are some other emulators, for instance, OpenvSwitch. It is an emulator widely used to test experiments about networking, emulating an OpenFlow software-based switch. In [28] Akella and Xiong made a study about QoS in SDN networks. They presented a bandwidth allocation approach by using Open vSwitch. Their study is useful for most of cloud applications like gaming, voice IP and teleconference. They achieved the guarantee bandwidth allocation to all cloud users by introducing queuing techniques and considering the performance metrics of response time and the number of hops.

On the other hand, other tools like GNS3 are used to emulate the SDN designed in the experiments. For example, in [29] Jingjing et al. researched the deployment of routing protocols over SDN emulated by GNS3. They used and optimized an architecture called Kandoo, a distributed control plane architecture, to enhance routing in SDN. They also analyze BGP and OSPF routing protocols and concluded their routing strategies are superior to the traditional ones based on BGP and OSPF. They used for the simulation GNS3 emulator, which is essential to evaluate their results and finalizing the research.

Other test bed used in several researches is the OFELIA project, based in OpenFlow. It is an experimental SDN designed to research about networking in SDN. In [30] Salsano et al. discussed and proposed a general and long term solution to support ICN (Information Center Networking) over a large scale SDN based on Openflow using OFELIA to experimenting their proposal.

Furthermore, networks designed in order to simulate the experiments, like OFELIA, provide new opportunities to develop other tools like VeRTIGO, expanding the SDN possibilities over any topology.

Gerola et al. tested VeRTIGO [31] to demonstrate this new tool has the power of allowing investigators work with OpenFlow-based SDN in a large scale in terms of topology. VeRTIGO has been developed within the framework OFELIA project.

Beside the emulators, which can virtualize networks in order to test the proposals, there are simulators, which try to simulate components and network behavior. Usually, emulators achieve a better performance than simulators. In [32] Wang et al. introduced EstiNet, a new tool to make experiments which consists on a mix of emulator and simulator. They presented it for testing performance of SDN OpenFlow controller's application programs. Without any modification, they could run OpenFlow controllers into an EstiNet virtual network. They concluded that EstiNet, by combining simulation and emulation, take the advantages of both approaches and it is able to avoid their disadvantages. Finally, they made a comparison between EstiNet, Mininet and ns-3 simulator, concluding EstiNet is even better than Mininet because it is more scalable, although it takes more time simulating more OpenFlow switches, and generates correct performance, while Mininet performance and results are untrustworthy. This comparison is detailed in [33], written by Wang.

III. TEST BENCH

In this section, we are going to introduce the SDN emulator and the real network topology used in our test bench.

A. Devices and equipement

In this subsection, we explain the devices and equipment used to perform our study.

The real topology is composed by the following equipment:

- 1 Layer 3 Switch, Cisco Catalyst WS-C3560-24PS-E [34] that runs an IOS C3560-IPSERVICESK9-M,Versión 12.2 (53) SE2, release software (fc3). It

has 24 Fast Ethernet and 2 Gigabit Ethernet interfaces and 16 Mbytes of flash memory;

- 1 Desktop PC that has an Intel Core Quad Q9400 CPU @2.66 Ghz processor, 6 Gb of RAM memory, 1 Network Interface Card (NIC) Intel 82579V Gigabit Ethernet and Windows 7 Professional - 64 bits operative system;
- • 1 Desktop PC that has an Intel Core i5-2400 CPU @3.10 Ghz, 4 Gb RAM memory, 1 NIC Intel 82579V Gigabit Ethernet and Windows 7 Enterprise - 64 bits as operating system.

To design and develop the virtualized topology we have used a laptop composed by an Intel i7-4500UCPU @ 2.70 Ghz processor, 16 Gb RAM memory, 1 10/100/1000 Mbit/s NIC, and Ubuntu 14.04 - 64 bits as operating system.

B. Software used

With Mininet, we can create a realistic virtual network, running real kernel, switch and application code, on a single machine. The machine can be a virtual machine running on a local PC, or a machine virtualized through the cloud, or a native machine. For our study, we have used Mininet version 2.2.1, with a native installation on Ubuntu 14 as shown in Figure 2.

We used a software application named gt, programmed with a C Linux compiler and developed by us, which allow us to send traffic with different MTU and bandwidths set by the user. Varying the frame interdelay and frame size, it is easy to get any desired speed, as far as it not higher than the physical interface speed. The components of the transport line are not significant four these tests

In both, real and virtualized topologies, to capture and analyze the received traffic, we have used Wireshark [35], version 1.10.

C. Characteristics of traffic transmited

In our work, we send traffic with different MTUs that represents the packet sizes in different standards. Table I shows different sizes of MTU that was sent in our network topologies.

As can be observed in Table I, sizes of MTU that was sent in our topology do not have standard values. This is because of the need to establish a GRE tunnel in the real topology, to connect the two hosts that have been created in Mininet, thus changing the frame size. Traffic was transmitted through UDP protocol. To calculate the jitter (J), we use the expression presented in RFC 4689 (Terminology for Benchmarking Network-layer Traffic Control Mechanisms) [36]. Therefore, we use the formula (1), where S_i is the transmission timestamp from packet i, and R_i is the reception timestamp of arrival packet i. For two consecutive packets i and j.

$$J = |(R_j - S_j) - (R_i - S_i)| \quad (1)$$

D. Physical topology

The real topology consists of two computers connected by straight-through cable, using one real switch (Cisco Catalyst WS-C3560-24PS-E), as shown in Figure 3. The data transfer rates used is 10 Mbps.

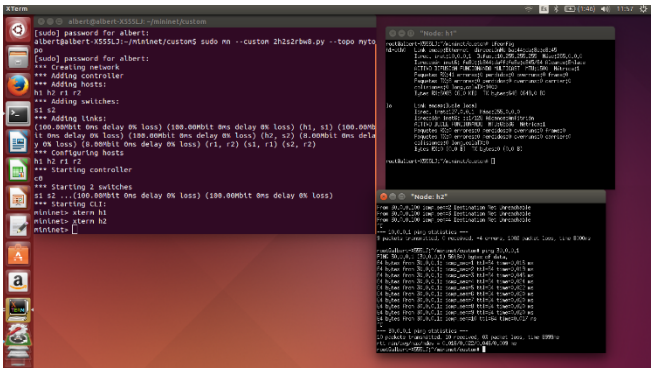

Figure 2. Host running in Mininet.

TABLE I. MTU PACKETS IN TOPOLOGIES

Frame Differentiation	
Media	**MTU (bytes)**
Ethernet wit LLC and SNP, PPPoE	1518
FDDI	4370
WLAN 802.11, Ethernet Jumbo Frame	7999

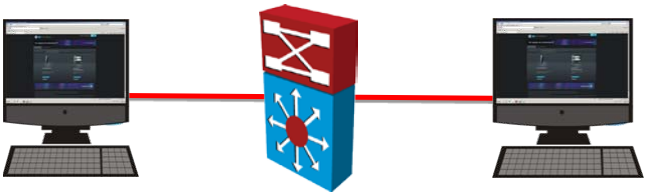

Figure 3. Real topology.

In the software defined network, we used a computer with Mininet, where we set up the same topology as the real one.

IV. MEASURAMENT AND DISCUSSION

This section shows the results obtained in both cases, when traffic is being delivered over the real network and in the virtual topology using Mininet. Here we present measures of traffic, when link bandwidth was configured at 10 and 100 Mbps, and the traffic generator was transmitting at 10 and 100 Mbps. Our intention is to test the ability of the devices to process packets of several sizes. For that, MTU of every interface is configured to allow those different packet sizes. The parameters observed are bandwidth and jitter of packets with three different MTUs: 1518, 4370 and 7999, corresponding at size of packets for traffic Ethernet, FDDI and WLAN 802.11.

The two Mininet networks running at different PCs were interconnected through a GRE tunnel established between both PCs. The GRE encapsulation is showed in Figure 4.

The experiments are described as follows. First, we group the experiments according to the maximum bandwidth available in the links of our topology. Then, we present the results obtained in words of bandwidth and jitter from both topologies, the real and the virtual one, for every MTU value we test.

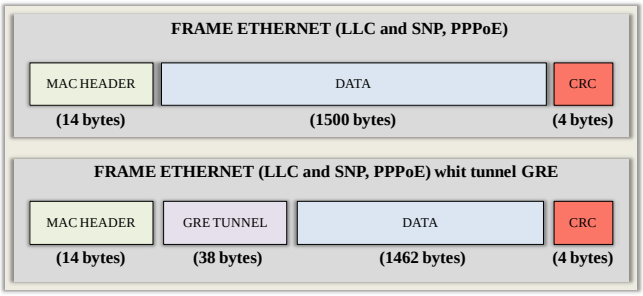

Figure 4. GRE tunnel.

1) Traffic links bandwidth 10 Mbps - Traffic generated 10 Mbps.

a) MTU - 1518

In Figure 5, we can see the bandwidth consumption values of the real topology and the values obtained in the virtual topology. The data have similar values for both topologies when the transmission is stabilized. Although, in real topology is less than in virtual topology. The mean value of bandwidth in real topology is 9.5 Mbps while for virtual topology is 10 Mbps. The maximum and minimum values for real and virtual topologies are different, 9.9 Mbps and 20.8 Mbps for maximum and 6.7 Mbps and 9.9 Mbps for minimum. Observe that in the virtual topology, at the beginning of the transmission we obtain bandwidth values higher than 10 Mbps, meaning that in this situation the emulator is not accurate since the maximum bandwidth for a emulated 10 Mbps physical link should be 10 Mbps. After a few transmitted packets, the measured bandwidth is already providing more accurate values.

In Figure 6, we can see the jitter values of the real topology and the values obtained in the virtual topology. The values of the real topology are higher than those from the virtual topology. The mean value of jitter in real topology is 0.690 ms while for virtual topology is 0.001 ms. The maximum values real and virtual topologies are different, 3.169 ms and 0.607 ms. The minimum values for both topologies are the same, 0 ms.

b) MTU - 4370

In Figure 7, we can see the bandwidth consumption values of the real topology and the values obtained in the virtual topology. The data have similar values for both topologies, when the transmission is stabilized, although in real topology is less than in virtual topology. The mean value of bandwidth in real topology is 9.5 Mbps while for virtual topology is 10 Mbps. The maximum and minimum values for real and virtual topologies are different, 9.8 Mbps and 31.5 Mbps for maximum, and 4.2 Mbps and 9.9 Mbps for minimum. As in the previous case, MTU 1518 bytes, in the virtual topology, we can observe that the bandwidth values are not realistic at the beginning of the transmission. After several transmitted packets, the values obtained are already close to the real network values.

In Figure 8, we can see the jitter values of the real topology and the values obtained in the virtual topology. The values of the real topology are higher than those from the virtual topology. The mean value of jitter in real topology is 0.228ms while for virtual topology is 0.002 ms. The maximum values for real topology are different, 9.189 ms and 1.277 ms. The minimum values for real topology and virtual topology are the same, 0 ms.

c) MTU - 7999

In Figure 9, we can see the bandwidth consumption values of the real topology and the values obtained in the virtual topology. The data have similar values for both topologies, when the transmission is stabilized, although in real topology is less than in virtual topology. The mean value of bandwidth in real topology is 9.5 Mbps while for virtual topology is 10 Mbps. The maximum and minimum values for real topology and virtual topology are different, 9.9 Mbps and 23 Mbps for maximum and 3.9 Mbps and 10 Mbps for minimum. Once again, the virtual topology is not providing realistic bandwidth values at the beginning of the transmission and, after transmitting a few packets, the bandwidth values are quite similar to those from the real network.

In Figure 10, we can see the jitter values of the real topology and the values obtained in the virtual topology. The values of the real topology are higher than those from the virtual topology. The mean value of jitter in real topology is 0.345 ms while for virtual topology is 0.001 ms. The maximum and minimum values for real topology and virtual topology are different, 25.091 ms and 0.844 ms for maximum and 0.037 ms and 0 ms for minimum.

As a conclusion, the packet size does not seem to have much impact on the observed bandwidth, but more on the jitter.

2) Traffic links bandwidth 100 Mbps - Traffic generated 10 Mbps.

a) MTU - 1518

In Figure 11, we can see the bandwidth consumption values of the real topology and the values obtained in the virtual topology. The data have similar values for both topologies, when the transmission was stabilized, although in real topology is less than in virtual topology. The mean value of bandwidth in real topology is 9,9 Mbps while for virtual topology is 10 Mbps. The maximum and minimum values for real topology and virtual topology are different, 10,0 Mbps and 10,3 Mbps for maximum and 9,9 Mbps and 9,8 Mbps for minimum.

In Figure 12, we can see the jitter values of the real topology and the values obtained in the virtual topology. The values of the real topology are higher than the ones of the virtual topology. The mean value of jitter in real topology is 0,015 ms while for virtual topology is 0,001 ms. The maximum values for real and virtual topology are different, 1,26 ms and 2,691 ms. The minimum values for real topology and virtual topology are the same, 0 ms.

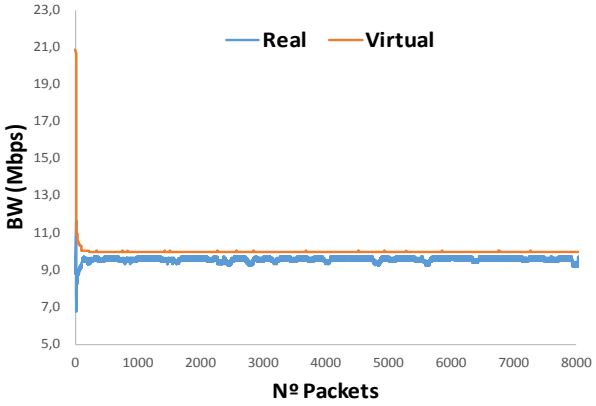

Figure 5. BW at 1518.

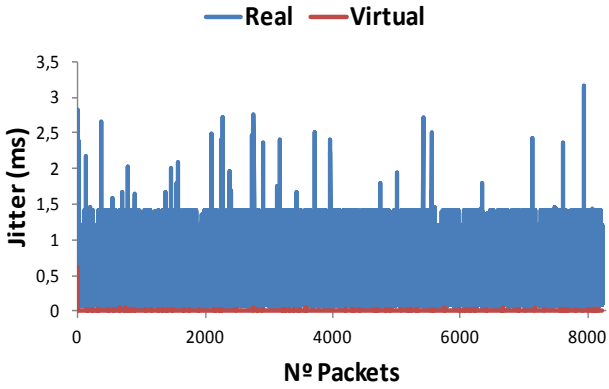

Figure 6. Jitter at 1518.

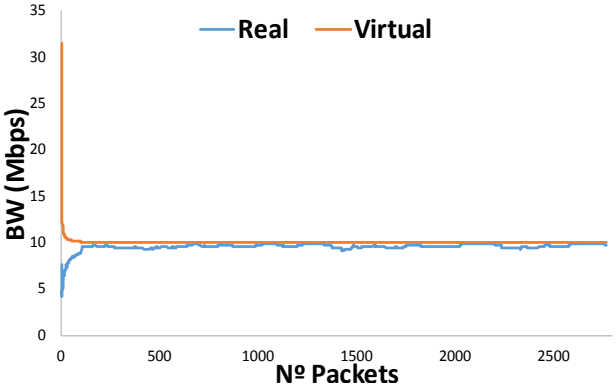

Figure 7. BW at 4370.

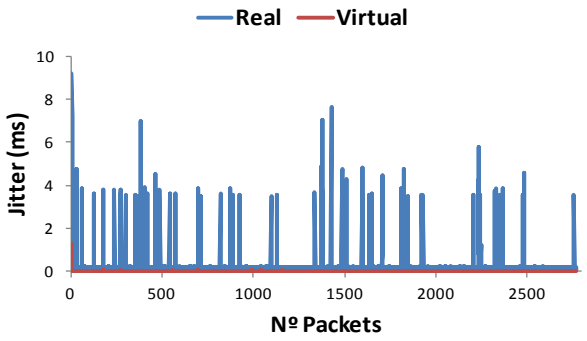

Figure 8. Jitter at 4370.

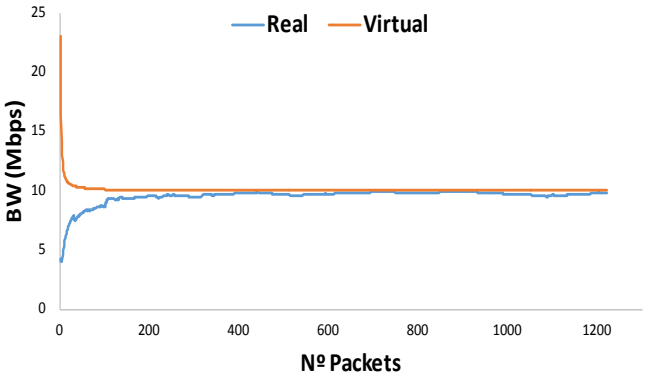

Figure 9. BW at 7999.

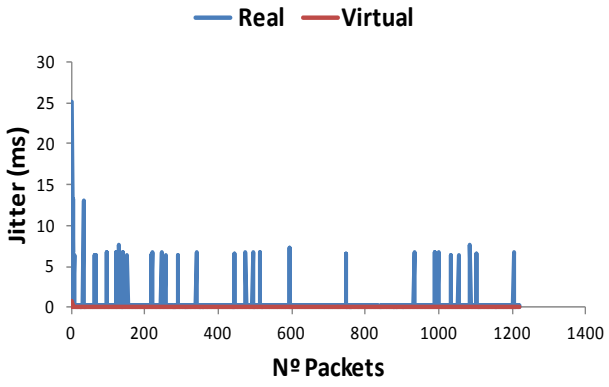

Figure 10. Jitter at 7999.

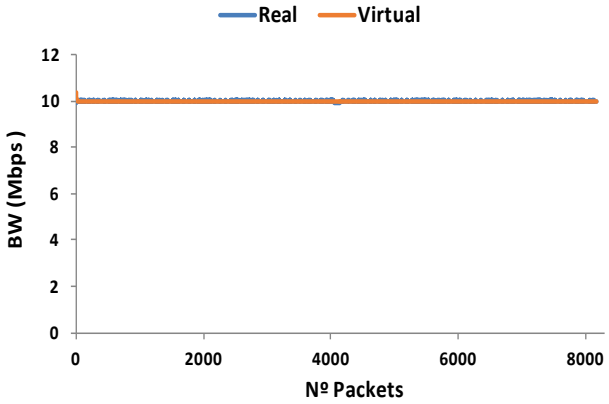

Figure 11. BW at 1518.

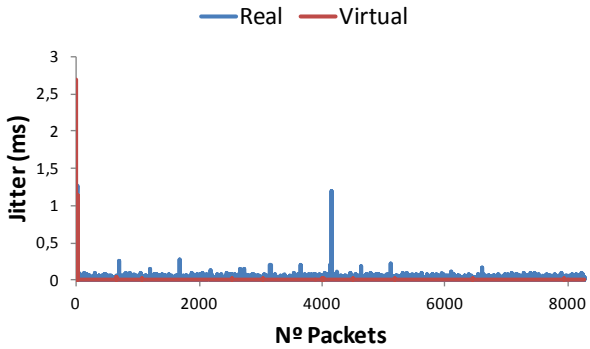

Figure 12. Jitter at 1518.

b) MTU - 4370

In Figure 13, we can see the bandwidth consumption values of the real topology and the values obtained in the virtual topology. The data have similar values for both topologies, when the transmission was stabilized. Although in real topology is less than in virtual topology. The mean value of bandwidth in real topology is 9.9 Mbps while for virtual topology is 10 Mbps. The maximum values for real and virtual topology are the same 10 Mbps. The minimum values for real topology and virtual topology are different 9.8 Mbps and 9.9 Mbps.

In Figure 14, we can see the jitter values of the real topology and the values obtained in the virtual topology. The values of the real topology are higher than the ones of the virtual topology. The mean value of jitter in real topology is 0.019 ms while for virtual topology is 0.004 ms. The maximum values for real topology and virtual topology are different, 3.433 ms and 0.898 ms. The minimum values for real topology and virtual topology are the same, 0 ms.

This section shows the results obtained in both cases, when traffic is being delivered over the network and in the virtual topology using Mininet.

c) MTU - 7999

In Figure 15, we can see the bandwidth consumption values of the real topology and the values obtained in the virtual topology. The data have similar values for both topologies, when the transmission was stabilized.. The mean value of bandwidth in real and virtual topology is 10 Mbps. The maximum and minimum values for real topology and virtual topology are the same, 10 Mbps for maximum and 9,9 Mbps for minimum.

In Figure 16, we can see the jitter values of the real topology and the values obtained in the virtual topology. The values of the real topology are higher than the ones of the virtual topology. The mean value of jitter in real topology is 0.021 ms while for virtual topology is 0.005 ms. The maximum values for real topology and virtual topology are different, 1.177 ms and 1.756 ms. These values occurred at the very beginning and the virtual peak hide the one related with the real topology. The minimum values for real topology and virtual topology are the same, 0 ms.

Sending less traffic than the maximum bandwidth available seems that introduces less discrepancy between the virtual topology and the real one.

3) Traffic links bandwidth 100 Mbps - Traffic generated 100 Mbps.

a) MTU - 1518

In Figure 17, we can see the bandwidth consumption values of the real topology and the values obtained in the virtual topology. The data have similar values for both topologies, when the transmission was stabilized, although in real topology is less than in virtual topology. The mean value of bandwidth in real topology is 96 Mbps while for virtual topology is 100 Mbps. The maximum and minimum values for real topology and virtual topology are different, 100 Mbps and 101 Mbps for maximum and 94.3 Mbps and 7.3 Mbps for minimum.

In Figure 18, we can see the jitter values of the real topology and the values obtained in the virtual topology. The values of the real topology are higher than then ones of the virtual topology. The mean value of jitter in real topology is 0.011 ms while for virtual topology is 0.001 ms. The maximum values for real topology and virtual topology are different, 0.25 ms and 3.26 ms. The minimum values for real topology and virtual topology are the same, 0 ms.

b) MTU - 4370

In Figure 19, we can see the bandwidth consumption values of the real topology and the values obtained in the virtual topology. The data have similar values for both topologies, when the transmission was stabilized, although in real topology is less than in virtual topology. The mean value of bandwidth in real topology is 98.559 Mbps while for virtual topology is 100 Mbps. The maximum and minimum values for real topology and virtual topology are different, 100 Mbps and 120.344 Mbps for maximum and 95.474 Mbps and 99.931 Mbps for minimum. Once again, the virtual topology is not providing realistic bandwidth values at the beginning of the transmission. But also, after transmitting a few packets the bandwidth values are similar to those from the real network.

In Figure 20, we can see the jitter values of the real topology and the values obtained in the virtual topology. The values of the real topology are higher than the ones of the virtual topology. The mean value of jitter in real topology is 0.017 ms while for virtual topology is 0 ms. The maximum values for real topology and virtual topology are different, 0,908 ms and 0.028 ms. The minimum values for real topology and virtual topology are the same, 0 ms.

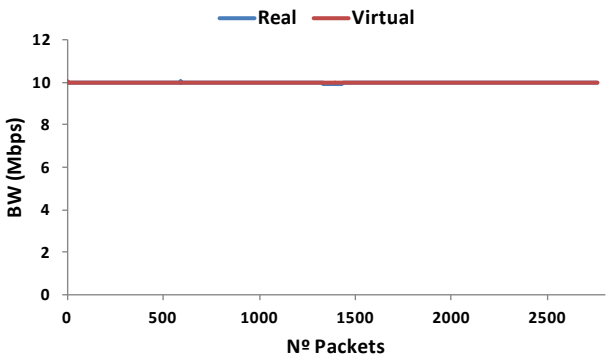

Figure 13. BW at 4370.

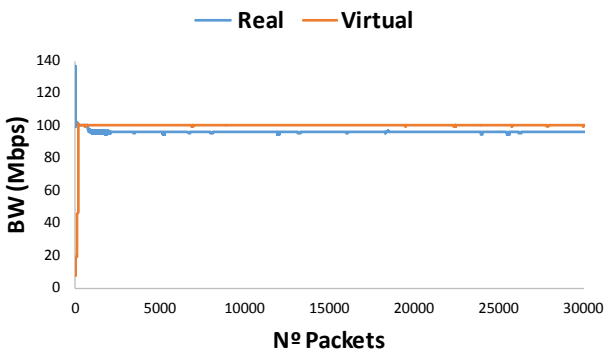

Figure 17. BW at 1518.

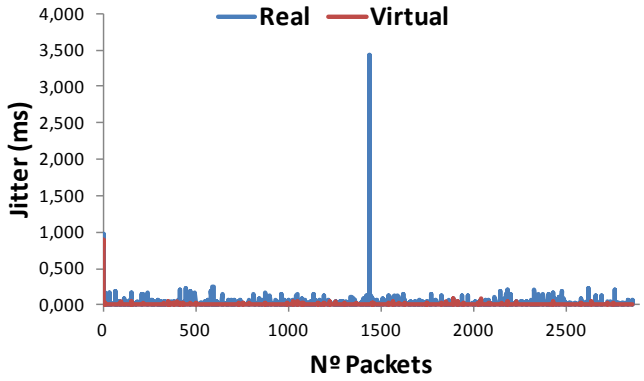

Figure 14. Jitter at 4370.

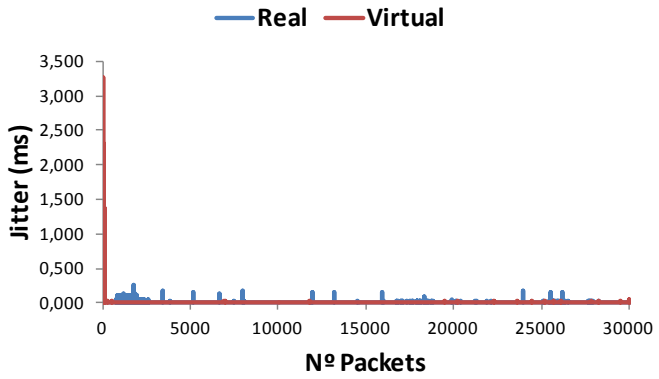

Figure 18. Jitter at 1518.

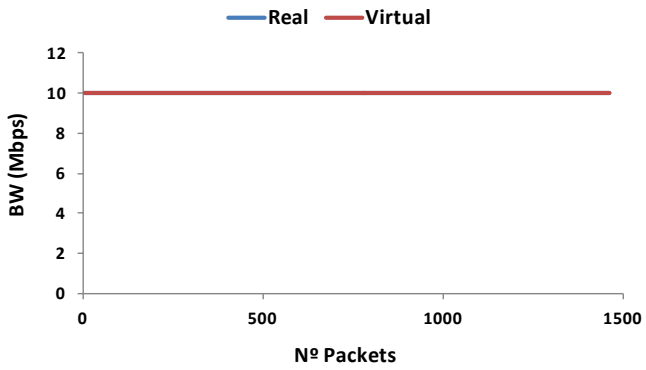

Figure 15. BW at 7999.

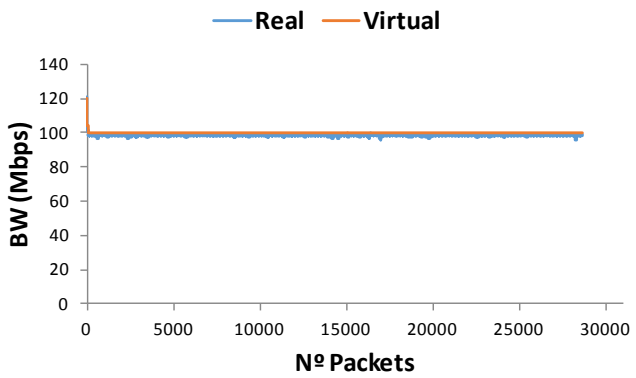

Figure 19. BW at 4370.

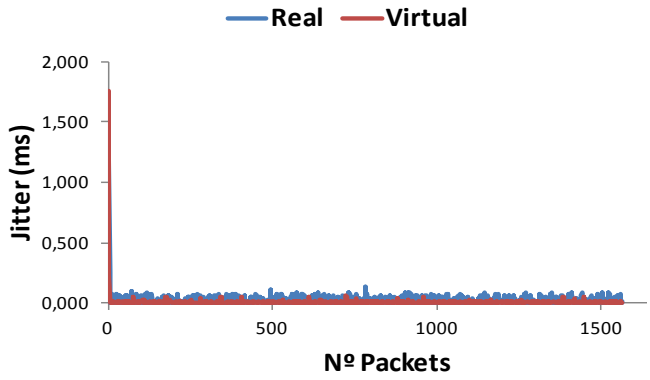

Figure 16. Jitter at 7999.

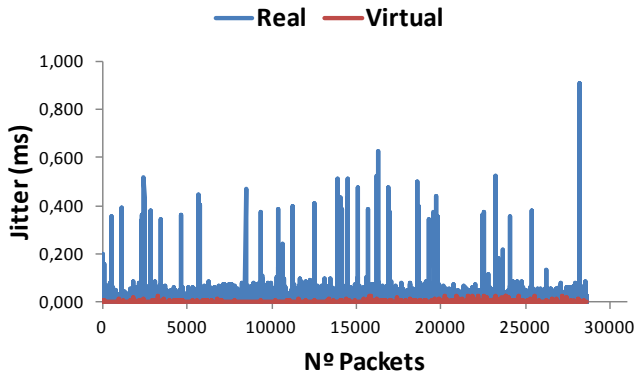

Figure 20. Jitter at 4370.

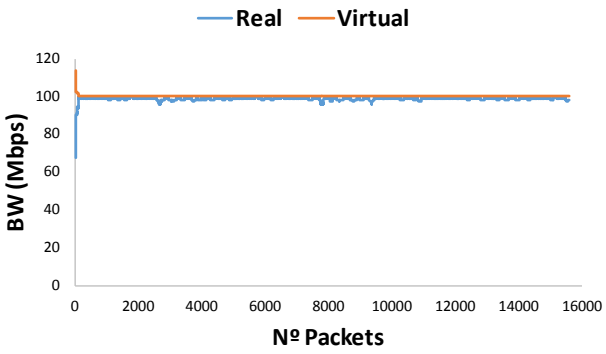

Figure 21. BW at 7999.

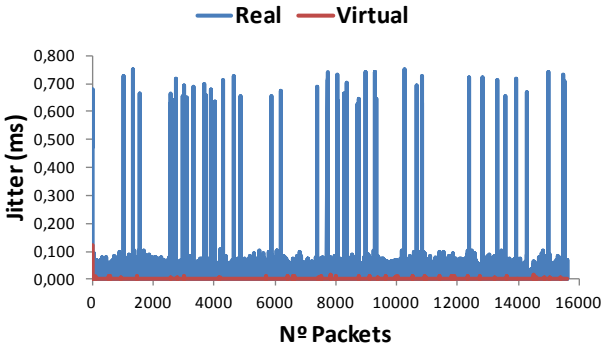

Figure 22. Jitter at 7999.MTU - 7999

This section shows the results obtained in both cases, when traffic is being delivered over the network and in the virtual topology using Mininet.

In Figure 21, we can see the bandwidth consumption values of the real topology and the values obtained in the virtual topology. The data have similar values for both topologies, when the transmission was stabilized. The mean value of bandwidth in real topology is 98.911 Mbps while for virtual topology is 100 Mbps. The maximum and minimum values for real topology and virtual topology are different, 99.502 Mbps and 113.581 Mbps for maximum and 67.687 Mbps and 100 Kbps for minimum (due to the initial interval that some devices need to start sending messages). As previously, virtual topology is not providing realistic bandwidth values at the beginning of the transmission.

In Figure 22, we can see the jitter values of the real topology and the values obtained in the virtual topology. The values of the real topology are higher than the ones of the virtual topology. The mean value of jitter in real topology is 0.016 ms while for virtual topology is 0 ms. The maximum values for real topology and virtual topology are different, 0.749 ms and 0.123 ms. The minimum values for real topology and virtual topology are the same, 0 ms.

These lasts measurements indicate that the virtual topology can consume more bandwidth than the available in the real network.

V. CONCLUSION

In this paper, we have studied the performance of virtual networks and compared with real networks. For this study, we have transmitted packets with different MTU sizes, which correspond to Ethernet, FDDI, and WLAN 802.11 (also Jumbo Ethernet frames) packets and we have used different link capabilities and traffic through the network. It can be seen that the variation of the bandwidth between the real and virtual topologies are very low. However, in virtual networks, the first packets are usually sent with an unreal bandwidth. The results obtained for the jitter show that there are major deviations, although, we are working with a very low time scale, as we are dealing with milliseconds. In our future work, we will compare real and virtual networks using more complex topologies, and with Openflow compatible equipment.

ACKNOWLEDGMENT

This work has been supported by the “Ministerio de Economía y Competitividad”, through the “Convocatoria 2014. Proyectos I+D - Programa Estatal de Investigación Científica y Técnica de Excelencia” in the “Subprograma Estatal de Generación de Conocimiento”, Project TIN2014-57991-C3-1-P and the “Programa para la Formación de Personal Investigador – (FPI-2015-S2-884)” by the “Universitat Politècnica de València”.

REFERENCES

[1] J. M. Jimenez, O. Romero, A. Rego, A. Dilendra, J. Lloret, Performance Study of a Software Defined Network Emulator, The Eleventh International Conference on Internet Monitoring and Protection (ICIMP 2016), May 22 - 26, 2016 - Valencia, Spain

[2] Omnet++. Available at https://omnetpp.org/ [Last access November 8, 2016]

[3] OPNET is now part of Riberved. Available at http://es.riverbed.com/products/performance-management-control/opnet.html / [Last access November 8, 2016]

[4] The Network Simulator - ns-2. Available at http://www.isi.edu/nsnam/ns/ [Last access November 8, 2016]

[5] NS-3. Available at NS-3 website: https://www.nsnam.org/ [Last access November 8, 2016]

[6] NetSim NETWORK SIMULATOR. Available at http://www.boson.com/netsim-cisco-network-simulator [Last access November 8, 2016].

[7] GNS3 The software that empowers networks professionals. Available at https://www.gns3.com/ [Last access November 8, 2016].

[8] Software-Defined Networking: A Perspective from within a Service Provider Environment. Available at: https:// tools.ietf.org/pdf/rfc7149.pdf [Last access November 8, 2016]

[9] Liboftrace. Available at http://archive.openflow.org/wk/index .php/Liboftrace [Last access November 8, 2016]

[10] OFLOPS. Available at https://www.sdxcentral.com/projects/ oflops/ [Last access November 8, 2016]

[11] OpenSeer. Available at http://archive.openflow.org/wk/ index.php/OpenSeer [Last access November 8, 2016]

[12] What are SDN Controllers (or SDN Controllers Platforms)?. Available at https://www.sdxcentral.com/resources/sdn/sdn-controllers/[Last access November 8, 2016]

[13] OpenFlow network virtualization with FlowVisor. Available at https://www.os3.nl/_media/2012-2013/courses/rp2/p28_report.pdf [Last access November 8, 2016]

[14] OpenFlow switch. Available at http://searchsdn.techtarget.com/definition/OpenFlow-switch [Last access November 8, 2016]

[15] Mininet An Instant Virtual Network on your Laptop (or other PC). Available at http://mininet.org [Last access November 8, 2016]

[16] EstiNet Technologies Inc. Available at EstiNet Technologies website: http://www.estinet.com/index.php [Last access November 8, 2016].

[17] J. M. Jimenez, O. Romero, A. Rego, A. Dilendra and J. Lloret, "Study of Multimedia Delivery over Software Defined Networking", in Network Protocols and Algorithm, vol. 7, No. 4, 2015, pp. 37-62, 2015, doi:10.5296/npa.v7i48794

[18] D. Kreutz, F. M. V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey", Proceedings of the IEEE, Volume103, Issue 1, Jan. 2015, pp. 14-76, 2015, http://dx.doi.org/10.1109/JPROC.2014.2371999

[19] K. A. Noghani and M. O. Sunay, "Streaming Multicast Video over Software-Defined Networks", Proceedings of the IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems (MASS) (2014), 28-30 Oct. 2014, pages 551-556, 2014, doi: 10.1109/MASS.2014.125

[20] H. Nam, K. Kim, J. Y. Kim and H. Schulzrinney, "Towards QoE-aware Video Streaming using SDN", Global Communications Conference (GLOBECOM), Dec. 2014, pp 1317-1322, 2014, doi: 10.1109/GLOCOM.2014.7036990

[21] H. E. Egilmez, S. T. Dane, K. Tolga Bagci and A. Murat Tekalp, "OpenQoS: An OpenFlow controller design for multimedia delivery with end-to-end Quality of Service over Software-Defined Networks", Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific, 3-6 Dec. 2012, Hollywood (USA), pp 1-8, 2012

[22] H. E. Egilmez, and A. M. Tekalp, "Distributed QoS Architectures for Multimedia Streaming Over Software Defined Networks", Multimedia, IEEE Transactions on , Volume:16 , Issue: 6 , Sept. 2014, pages: 1597 – 1609, 2014; doi:10.1109/TMM.2014.2325791

[23] A. Kassler, L. Skorin-Kapov, O. Dobrijevic, M. Matijasevic, and P. Dely, "Towards QoE-driven Multimedia Service Negotiation and Path Optimization with Software Defined Networking", Software, Telecommunications and Computer Networks (SoftCOM), IEEE, Sept. 2012, Split (Croatia), pages: 1-5, 2012, ISBN: 978-1-4673-2710-7

[24] H. Yang, J. Zhang, Y. Zhao, Y. Ji, J. Han, Y. Lin, S. Qiu, Y. Lee, Time-aware Software Defined Networking for OpenFlow-based Datacenter Optical Networks, Network Protocols and Algorithms, Vol 6, No 4 (2014). pp. 77-91.

[25] M. Dramitinos, N. Zhang, M.. Kantor, J. Costa-Requena, I. Papafili, "Video Delivery over Next Generation Cellular Networks", Network and Service Management (CNSM), 2013 9th International Conference, IEEE, 14-18 Oct. 2013, Zurich (Switzerland), pp. 386-393, doi> 10.1109/CNSM.2013.6727862t

[26] R. L. S. de Oliveira, A. A. Shinoda, C. M. Schweitzer, L. Rodrigues Prete, "Using Mininet for Emulation and Prototyping Software-Defined Networks", Communications and Computing (COLCOM), 2014 IEEE Colombian Conference, IEEE, 4-6 June 2014, Bogota (Colombia), pp. 1 - 6, DOI 10.1109/ColComCon.2014.6860404

[27] P. Wette, M. Dräxler, A. Schwabe, F. Wallaschek, M. H. Zahraee, H. Karl, "MaxiNet: Distributed Emulation of Software-Defined Networks", Networking Conference, 2014 IFIP, 2-4 June 2014, Throndheim (Norway), pp. 1-9, DOI 10.1109/IFIPNetworking.2014.6857078

[28] A. V. Akella, K. Xiong, "Quality of Service (QoS) Guaranteed Network Resource Allocation via Software Defined Networking (SDN)", Dependable, Autonomic and Secure Computing (DASC), 2014 IEEE 12th International Conference, 24-27 Aug. 2014, IEEE, Dalian (China), pp. 7-13, DOI 10.1109/DASC.2014.11.

[29] Z. Jingjing, C. Di, W. Weiming, J. Rong, W. Xiaochun, "The Deployment of Routing Protocols in Distributed Control Plane of SDN", in The Scientific World Journal, Volume 2014, Article ID 918536, 8 pages, http://dx.doi.org/10.1155/2014/918536

[30] S. Salsano, N. Blefari-Melazzi, A. Detti, G. Morabito, L. Veltri, "Information centric networking over SDN and OpenFlow: Architectural aspects and experiments on the OFELIA testbed", Computer Networks, Volume 57, Issue 16, 13 November 2013, pp. 3207-3221, DOI 10.1016/j.comnet.2013.07.031

[31] M. Gerola, R. Doriguzzi Corin, R. Riggio, F. De Pellegrini, E. Salvadori, H. Woesner, T. Rothe, M. Suñé, L. Bergesio, "Demonstrating in ter-testbed network virtualization in OFELIA SDN experimental facility", Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference, 14-19 April 2013, Turin (Italy), pp. 39-40, DOI 10.1109/INFCOMW.2013.6970724

[32] S.-Y. Wang, C.-L. Chou, C.-M. Yang, "EstiNet OpenFlow Network Simulator and Emulator", in Communications Magazine, IEEE (Volume:51 , Issue: 9), IEEE, September 2013, pp. 110-117, DOI 10.1109/MCOM.2013.6588659

[33] S.-Y. Wang, "Comparison of SDN OpenFlow Network Simulator and Emulators: EstiNet vs. Mininet", in Computers and Communication (ISCC), 2014 IEEE Symposium, 23-26 June 2014, Funchal (Portugal), pp 1-6, DOI 10.1109/ISCC.2014.6912609

[34] Cisco Catalyst 3560 Series Switches Data Sheet. Available at Cisco website: http://www.cisco.com/c/en/us/products/collateral/switches/catalyst-3560-series-switches/product_data_sheet09186a00801f3d7d.html [Last access January 14, 2016]

[35] Wireshark software. Available at Wireshark website: https://www.wireshark.org/ [Last access January 14, 2016]

[36] Terminology for Benchmarking Network-layer Traffic Control Mechanisms. Available at https://www.ietf.org/rfc/rfc4689.txt.txt [Last access January 14, 2016]