Study of Multimedia Delivery over Software Defined Networks

Jose M. Jimenez, Oscar Romero, Albert Rego, Avinash Dilendra, Jaime Lloret

Universidad Politécnica de Valencia

Camino Vera s/n, 46022, Valencia (Spain)

E-mail: jojiher@dcom.upv.es, oromero@dcom.upv.es, alrema91@gmail.com, avinash.dilendra@gmail.com, jlloret@dcom.upv.es

Received: November 28, 2015 Accepted: December 31, 2015 Published: December 31, 2015

DOI: 10.5296/npa.v7i4.8794 URL: http://dx.doi.org/10.5296/npa.v7i4.8794

Abstract

Software Defined Networks (SDN) have become a new way to make dynamic topologies. They have great potential in both the creation and development of new network protocols and the inclusion of distributed artificial intelligence in the network. There are few emulators, like Mininet, that allow emulating a SDN in a single personal computer, but there is lack of works showing its performance and how it performs compared with real cases. This paper shows a performance comparison between Mininet and a real network when multimedia streams are being delivered. We are going to compare them in terms of consumed bandwidth (throughput), delay and jitter. Our study shows that there are some important differences when these parameters are compared. We hope that this research will be the basis to show the difference with real deployments when Mininet is used.

Keywords: Multimedia delivery; Multimedia streaming; Software Defined Networks (SDNs); Mininet.

1. Introduction

Due to the substantial improvement in terms of hardware and software for computer networks and also the number of network devices available around the world, the interconnections of devices as well as the network complexity have increased. This fact has also enhanced the way we currently process the information transmitted through the network. Over the past 30 years, the Internet Engineering Task Force (IETF) has developed and built around 5500 Request for Comments (RFC) [1]. Nowadays, Internet and other networks are able to offer huge amount of functionalities suited to the requirements of users.

In general, network devices perform two main functions, i.e., the transport function and the control function. On the one hand, the transport function (data plane) that is in charge of sending data through the routes previously calculated. This function is normally performed by specialized circuits known as Application Specific Integrated Circuits (ASICs). The control function (control plane) manages the transport operation thanks to the exchanged information between network devices and the calculation of optimal routes. This allows each device to independently treat the traffic. Network administrators have available few resources to manage and increase the efficiency of the data flows.

A professional of networks often finds a big challenge the fact of configuring a network and installing the needed network elements to work properly. Due to the services and the features required by the applications currently require, it is possible to increase the network efficiency if we try to manage jointly the entire network. In this way, there appears the need of developing a new technology to reduce the costs and increase the efficiency by automating policy-based flows.

Software-Defined networking (SDN) is a new approach for designing, building, and managing networks that separates the network’s control and forwarding planes of a better network optimization [2]. The SDN architecture decouples the network control and forwarding functions, allowing the network control to directly become a programmable and the underlying infrastructure to be abstracted from applications and network services [3]. SDN architecture is directly configured programming, agile in its operation, and based on a centralized management. In addition, SDN is open standards-based and vendor-neutral. Figure 1 show SDN architecture.

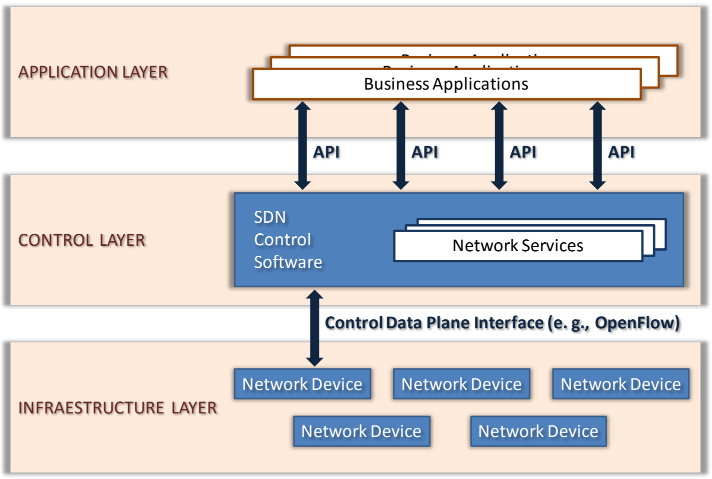


Figure 1: SDN Architecture

IETF defines the Software-Defined Networking as the set of techniques used to facilitate the design, delivery and operation of network services in a deterministic, dynamic, and scalable manner [4]. We can differentiate two different SDN approaches taking into account their behavior. These are the Open SDN and the Network Function Virtualization (NFV).

Open SDN uses the OpenFlow protocol [5]. OpenFlow is an open standard that enables researchers to run experimental protocols in campus networks. As a result, enterprises and network operators gain unprecedented programmability, automation, and network control, enabling them to build highly scalable, flexible networks that readily adapt to changing business needs [6]. OpenFlow protocol is a foundational element for building SDN solutions.

In NFV, the network devices such as routers, firewalls and load balancers, among others run virtualized on commodity hardware. In November 2012, ETSI created an Industry Specification Group for NFV [7]. Their mission was to facilitate the industry transformation and development of an open, interoperable ecosystem through specification, implementation and deployment experience.

In current networking world, SDN has many advantages. Firstly, SDN centralizes and simplifies the control of the enterprise network management. In addition, SDN offers to network owners and operators more control over their infrastructure, allowing customization and optimization for reducing the overall capital and operational costs. It allows the service providers to generate new revenue opportunities at an accelerated pace through the creation of software based applications for personal computers (PCs), mobile, and Web industries.

As a summary, the most important benefits of SDN is the cost reduction, network overhead reduction, improvements in network management, the decrease of network downtime and the extension of the capabilities of a SDN solution by developing applications to control the behavior of the networking traffic.

SDN has suffered a huge expansion in the last 3 years. Initially, SDN centered its development on the Openflow protocol which emerged firstly in campus environments and data centers. But it was rapidly migrated and adapted to the WAN. Recently, network operators and researchers are including network virtualization in their deployments and research lines. As a consequence of this, nowadays there are many publications about real implementation and proposals [8].

Because of all these reasons and the great evolution of SDN for developing new network applications, in this paper, we perform a comparative study to analyze the accuracy in terms of performance of a real network and a network implemented with Mininet. To do this, we have tested the transmission of multimedia streaming using both real and virtualized devices. Transmissions have been performed over UDP protocol, using the Real-Time Transport Protocol (RTP) [9]. The virtualization of network devices and hosts has been carried out with Mininet [10]. Mininet is a highly custom flexible SDN emulator that supports the OpenFlow protocol. Our goal is to compare the results between real and virtualized devices, to verify the accuracy of the results of multimedia delivery in virtualized environments.

The rest of the paper is structured as follows. In Section 2, we discuss existing related works. In Section 3, we introduced all resources that we used in our test bench. Measurement results and our discussion and analysis are shown in Section 4. Section 5 shows the conclusion and future works.

2. Related works

This section shows some published works introducing SDN and providing the results of implemented schemes and network topologies.

There are some published works that help understand the reader the concept of SDN and OpenFlow and provide some network implementations [11]. Hu et al. compiled the most important topics about SDN implementation and performed a comparison of different SDN schemes, discussing the future of this research area. Authors focused their work on how SDN separate data and control planes, making scalable networks developments based on SDN. Scott-Hayward et al. [12] highlighted the need of designing security schemes in SDN. Authors classified the security challenges attempting to which SDN layer are affected by. In addition, they proposed some solutions to these challenges and made emphasis on the importance of further work in order to achieve a secure and robust SDN environment, using the capability of being programmed and allowing a centralized network.

Kreutz et al. [13] presented a comprehensive overview of SDN and, despite its fundaments are not new, how SDN could be the new and revolutionary paradigm for Internet. Thus, they did a complete analysis of SDN infrastructure and the main challenges of SDN.

Kaur et al. [14] presented Mininet, an emulator for deploying networks on a single Virtual Machine as a method to make cheaper tests than tests over physical devices. Authors said that Mininet offered an easy use, performance accuracy and scalability for developing networks. Actually, Mininet has become a powerful tool which makes it easier to test complex networks without the need of having real hardware. In this sense in [15], Paasch et al. evaluated some scenarios for TCP multipath which could derive in huge costs in material like OpenFlow switches.

We can also find some previous works about experiments and performance tests of SDN and Mininet. De Oliveira et al. [16] explained the SDN paradigm, its elements, its purpose and its structure. They also presented Mininet and how it can help researchers to avoid the use of real and expensive hardware for networks. Authors exposed the need of making it easier to implement networks and how Mininet achieves it, with good performance and great reliability.

Moreover, Keti et al. [17] presented an evaluation of Mininet to study its limitations. The results showed that the simulation environment can generate a remarkable effect on the required time for implementing a topology.

In [18], Azizi et al. proposed a new model for measuring the delay in SDN (using Mininet), on a Multiprotocol Label Switching - Transport Profile (MPLS-TP) network. Authors proved that Mininet is not a good emulator for a stress test. However, SDN and Mininet can be used to get other measures as delay values. For instance, Gupta et al. [19] described the design, implementation and the use of fs-sdn (a simulation-based tool) to solve the problem of prototyping and accurately evaluating, at large scale, new SDN-based applications. The results enable the easy translation from virtual environment to real controller platforms like POX and NOX. Authors used Mininet in nearly identical configurations to compare it with fs-sdn.

Panwaree et al. [20] presented an evaluation of the network performance of video streamin over two kinds of OpenFlow-enabled network testbeds. They show the measurements obtained of delay and packet loss when video is streamed over Mininet and Open vSwitch, emulating a network, installed in a PC.

Furthermore, Megyesi et al. [21] introduced a novel mechanism for measuring available bandwidth in SDN networks. They built an application over the Network Operating System (NOS) which was able to track the network topology and the bandwidth utilization over the network links, and thus the mechanism was able to calculate the available bandwidth between any two points in the network. Authors validated their method using Mininet network emulation environment.

We have seen some deployments that use SDN based testbeds for multimedia streaming.

Noghani et al. [22] proposed a SDN-based IP multicast framework in order to control a set of video sources and the impact of SDN on Quality of Experience (QoE). This framework significantly increased the PNSR of the video, so the user received a good-quality video from a video almost impossible to see. Another practical case is shown by Michael Jarschel et al. [23]. They used a video streaming service like Youtube, in order to evaluate the performance over SDN see how OpenFlow can help us to improve the Quality of Service (QoS) in video delivery applications.

SDN networks can also be mixed with IP networks. Salsano et al. [24] described the design and implementation of an Open Source Hybrid IP/SDN (OSHI) node. They provided a set of open source tools that allowed facilitating the design of hybrid IP/SDN experimental networks. Their work shows the deployment on Mininet and on distributed SDN, and their test bench. In this way, it is possible to evaluate the costs related with the SDN integration on the Internet.

As we have seen, there are many researches related on Mininet and SDN performance. However, as far as we known, there are very few works showing a performance comparison between real transport protocol over a real network and in a SDN with Mininet is presented. The most similar work is presented in [25], where the Lantz et al. determined that Mininet was not accurate enough by making an experiment. The real hardware network gave different performance than the one emulated by Mininet. This difference was bigger with high network loads due to Linux scheduler, although the authors thought that this limitation is not very important.

In this paper, we are going to quantitatively determine these differences in order to know the grade of realism that Mininet can offer to emulate networks.

3. Test bench

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

3.1 Devices and equipment

This subsection describes the devices and equipment used to perform our study.

The real topology is composed by the following equipment:

- 2 Router Cisco 2811 Series [26], that runs an IOS C2800NM-ADVIPSERVICESK9-M, Version 15.0 (1) M, Release Software (fc2).

It has 2 Fast Ethernet and 2 Serial (sync/async) interfaces and 64 MBytes of Compact Flash;

- 2 Switches Cisco Catalyst WS-C3560-24PS-E [27] 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.

In order to connect the PCs and network devices, we have used:

- Cables UTP Cat.5e to connect the NICs (1.5 meters);
- Cisco CAB-SS-V35MT Smart Serial DTE/DCE WAN Cable serial cable to connect WAN interfaces of routers.

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.

3.2 Software used

In addition to the operating systems, we have used some software applications. We used, on the one hand, a software application to send and receive multimedia streaming and, on the other hand a software application to capture the received packets on the target PC.

In the real topology, VLC [28], version 2.0.8 Twoflower has been used to stream video from the server and in the client to receive the video streaming. To capture and analyze the received traffic, we have used Wireshark [29], version 1.10.5.

In the virtualized topology, we have used VLC version 2.16 for both, send and receive video streams and Wireshark version 1.10.6 to gather data.

3.3 Video and multimedia streaming characteristics

We have used a video with the following characteristics: 50 seconds length, 800X600 frame size, 1026 kbps bit rate (BRV) and 60 frames per second.

To send the multimedia traffic, we have used a Real-Time Transport Protocol (RTP) specified by the IETF through the RFC 1889 [9]. This was designed by the IETF's Audio-Video Transport Working Group to support video conferences with multiple geographically dispersed participants. RTP, by itself, does not guarantee the real-time delivery of multimedia data (since this is dependent on network characteristics); however, it provides the wherewithal to manage the data when it arrives. RTP is located above the UDP transport. The main function of RTP is to implement the sequence numbers of IP packets to reset the voice or video even when the underlying network changes the order of the packets.

The RTP header has a minimum size of 12 bytes. After the header, an optional header extension may be present. This is followed by the RTP payload, the format of which is determined by the particular class of application. Some underlying protocols may require a RTP encapsulation to be defined. Typically, one packet of the underlying protocol contains a single RTP packet, but several RTP packets may be contained if permitted by the encapsulation method. The main information of a RTP packet is the sequence number, timestamp and unique identifier for the source (SSRC).

In our study we send the RTP traffic through the UDP-port 5004. We need to configure this port in both PCs, source and destination, to properly work the multimedia streaming.

3.4 Physical topology

We have established six different cases in three topologies, for both real and virtual scenarios. This subsection describes the physical topologies we have used along the study when we work with the real equipment.

The first topology consists of two desktop PCs connected by a crossover cable, as it is shown in Figure 2. In this case we have sent traffic at 1 Gbps.

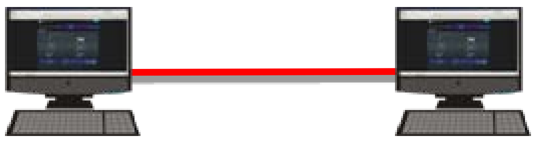


Figure 2. Physical Topology 1

The second topology consists of two desktop PCs connected, by straight-through cable, using 2 switches that are connected by a crossover cable, as it is shown in Figure 3. The data transfer rates used in this case has been 10 Mbps and 100 Mbps.

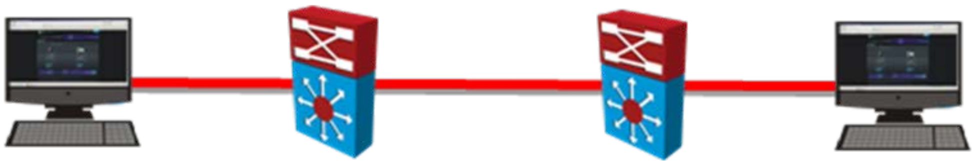


Figure 3. Physical Topology 2.

As Figure 4 shows, the third topology has added two routers between the switches. In this case, the link between routers has been a serial cable. The traffic has been sent at 2 Mbps, 4 Mbps and 8 Mbps through the WAN interfaces of the routers. The data transfer rates between PCs and switches is 100 Mbps.

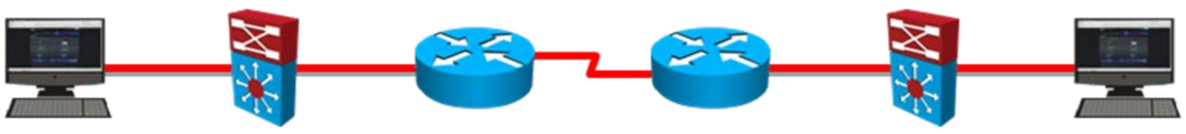


Figure 4. Physical Topology 3.

In the virtualized networks, we have used a PC with Mininet, in which we reproduced the same topologies that were employed with real equipment, as it is shown in Figure 5.

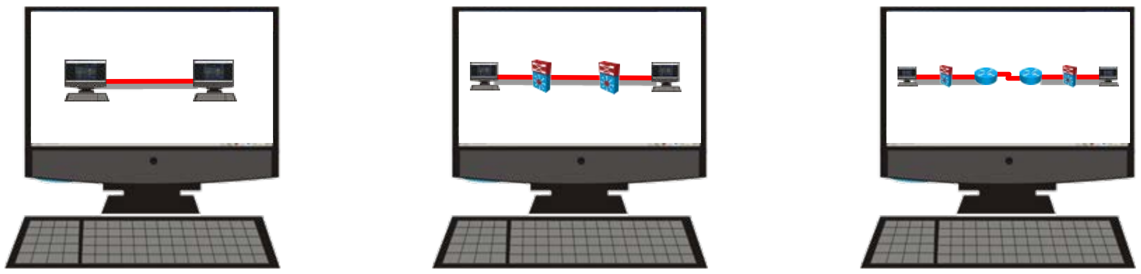


Figure 5. PC with virtualization used in all topologies.

3.5 SDN emulator: Mininet

There are several network simulators and emulators, but the most frequently used Simulators and Emulators for SDNs are the following ones:

- NS3 [30]: It is a C++ library which provides a set of network simulation models implemented as C++ objects and wrapped through python. It has OpenFlow support built in to emulate an Openflow environment and it can also be used for real-time simulations.
- EstiNet 9.0 [31]: It is a simulator and emulator can simulate thousands of Ver 1.3.4 and Ver 1.0.0 OpenFlow switches and run the real-world OpenDaylight [32], Ryu [33], NOX, and Floodlight [34] controllers without any modification to control these switches during simulation. Its performance simulation results are realistic, accurate, and repeatable. It supports both of the simulation mode and the emulation mode.
- Mininet: It is a network emulator which creates a network of virtual hosts, switches, controllers, and links. Mininet hosts run standard Linux network software, and its switches support OpenFlow for highly flexible custom routing and Software-Defined Networking. Version 2.2.1, installed natively on Ubuntu 14.04, can be used as a rapid prototyping for SDN. Mininet enables researchers to quickly create, interact with, customize and share a software defined network prototype, and provides a smooth path to running on hardware.

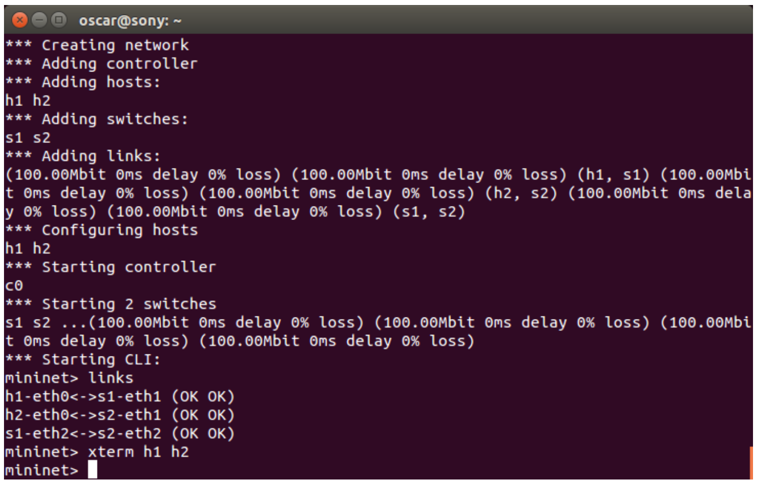
With Mininet, we can create a realistic virtual network, running real kernel, Switch and application code, on a single machine. The machine can be a Virtual Machine, or a machine virtualized through the cloud or native. For our study we have used Mininet version 2.2.1, with a native installation on Ubuntu 14.

Network topologies can be created using the commands: self.addHost( ) for adding a host, self.addSwitch( ) for adding a switch, self.addNode( ) for addinng a Linux router, and self.addLink( ) to establish links between hosts, switches and routers. To change the link speed, we use the optional parameter ‘bw=xx’, where xx is the speed in Mbps. An example of the code to set up a network topology with 2 PCs and 2 switches with default link speed is shown in Figure 6.

```
# Add hosts and switches
h1 = self.addHost( 'h1' )
h2 = self.addHost( 'h2' )
s1 = self.addSwitch( 's1' )
s2 = self.addSwitch( 's2' )
# Add links
self.addLink( h1, s1 )
self.addLink( h2, s2 )
self.addLink( s1, s2 )
```

Figure 6. Example of code to add hosts and switches in Mininet.

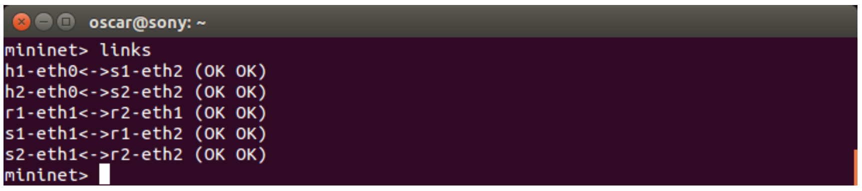
Figure 7 shows the result of defining the topology previously defined, with all links at 100 Mbps. Links between hosts and switches are also shown in the Mininet command line.



```
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2
*** Adding links:
(100.00Mbit 0ms delay 0% loss) (100.00Mbit 0ms delay 0% loss) (h1, s1) (100.00Mbi
t 0ms delay 0% loss) (100.00Mbit 0ms delay 0% loss) (h2, s2) (100.00Mbit 0ms dela
y 0% loss) (100.00Mbit 0ms delay 0% loss) (s1, s2)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...(100.00Mbit 0ms delay 0% loss) (100.00Mbit 0ms delay 0% loss) (100.00Mbi
t 0ms delay 0% loss) (100.00Mbit 0ms delay 0% loss)
*** Starting CLI:
mininet> links
h1-eth0<->s1-eth1 (OK OK)
h2-eth0<->s2-eth1 (OK OK)
s1-eth2<->s2-eth2 (OK OK)
mininet> xterm h1 h2
mininet>
```

Figure 7. Launch topology.

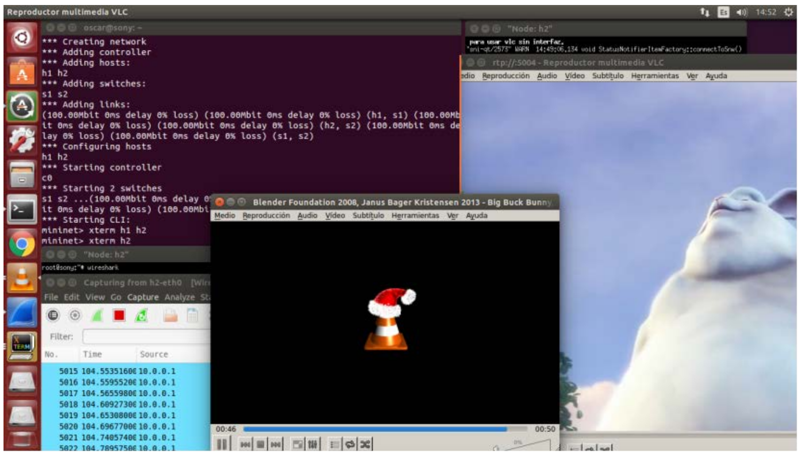
Figure 8 shows the links for topology formed with 2 routers and 2 switches (scenario of topology 2).



```
mininet> links
h1-eth0<->s1-eth2 (OK OK)
h2-eth0<->s2-eth2 (OK OK)
r1-eth1<->r2-eth1 (OK OK)
s1-eth1<->r1-eth2 (OK OK)
s2-eth1<->r2-eth2 (OK OK)
mininet>
```

Figure 8. Links, topology with 2 routers and 2 switches.

After defining the network topology within Mininet, next step is to send a video from host h1 to host h2. To this aim, we have used VLC version 2.1.6 for the video streaming in both, sender h1 and receiver h2. Thus, an Xterm session is launched from Mininet CLI for each host as shown in the Figure 2. In addition, we launch another Xterm session in the receiver in order to capture the received packets with Wireshark version 1.10.6. From this capture, we analyze the parameters of the video streaming for each situation. Figure 9 shows the desktop of a laptop when it is running Mininet with the network topology 2 and streaming the video from host h1 to host h2. The video is only displayed in the receiver and Wireshark captures the frames received in the network interface of host h2.



```
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2
*** Adding links:
(100.00Mbit 0ms delay 0% loss) (100.00Mbit 0ms delay 0% loss) (h1, s1) (100.00Mb
it 0ms delay 0% loss) (100.00Mbit 0ms delay 0% loss) (h2, s2) (100.00Mbit 0ms de
lay 0% loss) (100.00Mbit 0ms delay 0% loss) (s1, s2)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...(100.00Mbit 0ms delay 0
it 0ms delay 0% loss) (100.00Mbi
*** Starting CLI:
mininet> xterm h1 h2
mininet> xterm h2
```

```
root@sony:~# wireshark
```

```
No.   Time          Source
 5015 104.55351600 10.0.0.1
 5016 104.55955200 10.0.0.1
 5017 104.56559800 10.0.0.1
 5018 104.60927300 10.0.0.1
 5019 104.65308000 10.0.0.1
 5020 104.69677000 10.0.0.1
 5021 104.74057400 10.0.0.1
 5022 104.78957500 10.0.0.1
```

Reproductor multimedia VLC
oscar@sony: ~
"Node: h2"
Capturing from h2-eth0
File Edit View Go Capture Analyze
Filter:
Blender Foundation 2008, Janus Bager Kristensen 2013 - Big Buck Bunny
Medio Reproducción Audio Vídeo Subtítulo Herramientas Ver Ayuda
rtp://:5004 - Reproductor multimedia VLC
00:46
00:50
14:52

Figure 9. Screen capture where Mininet is running.

4. Measurements and discussion

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

4.1 Bandwidth

In this subsection we present the results related of bandwidth obtained for the three topologies.

4.1.1 Bandwidth consumed in topology 1

In Figure 10 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. The mean value of bandwidth in real topology is 1337.26 Kbps while for virtual topology is 1341.95 Kbps. The maximum values for real topology and virtual topology are the same 2592.67 Kbps. We obtained different minimum values, 162.72 Kbps for real topology and 86.78 Kbps for virtual topology (the initial 20 packets are not taken into account because the streaming was initiating). The variance of the data is 247128.43 for real topology and 244664.70 for virtual topology. The standard deviation is 497.12 in real topology and 494.63 in virtual topology. The data does not follow a normal distribution neither in real nor in virtual topology.

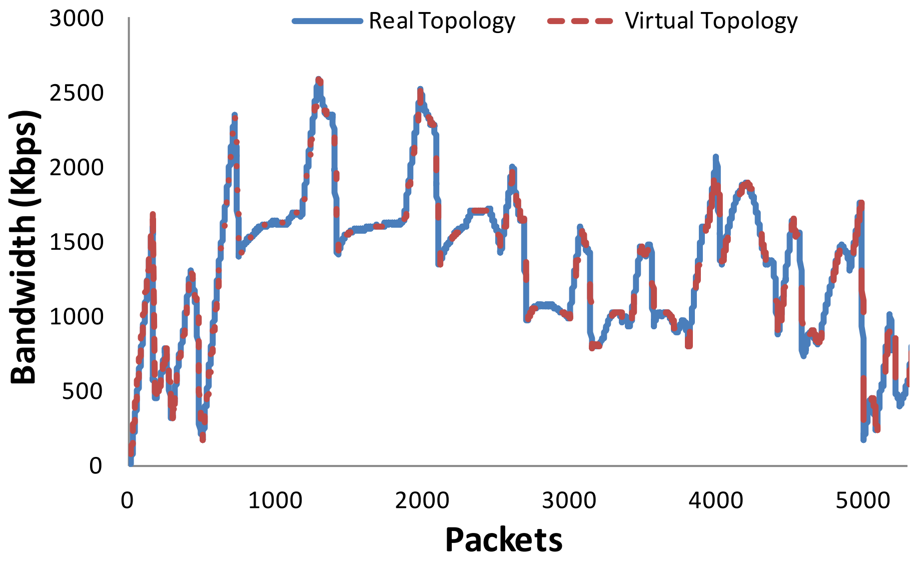


Figure 10. Bandwidth in topology 1, link at 1 Gbps.

4.1.2 Bandwidth consumed in topology 2

When we study the bandwidth consumption in topology 2, we can observe two different cases, when the link between switches is established at 10 Mbps and at 100 Mbps.

In Figure 11 we can see compared the values of bandwidth consumed in the real topology with the values of the virtual topology, when the link between switches is at 10 Mbps. The data have similar values in both topologies. The mean value of the bandwidth consumed in real topology is 1341.77 Kbps while for the virtual topology is 1351.85 Kbps. The maximum value for real topology and virtual topology are the same 2592.67 Kbps, but the minimum is slightly different, 162.72 Kbps for real topology and 173.57 Kbps for virtual topology (the initial 20 packets are not taken into account because the streaming was initiating). The variance of the data is 248874.16 for the real topology and 248150.91 for the virtual topology. The standard deviation is 498.87 in the real topology and 498.14 in the virtual topology. The data does not follow a normal distribution neither in real nor in virtual topology.

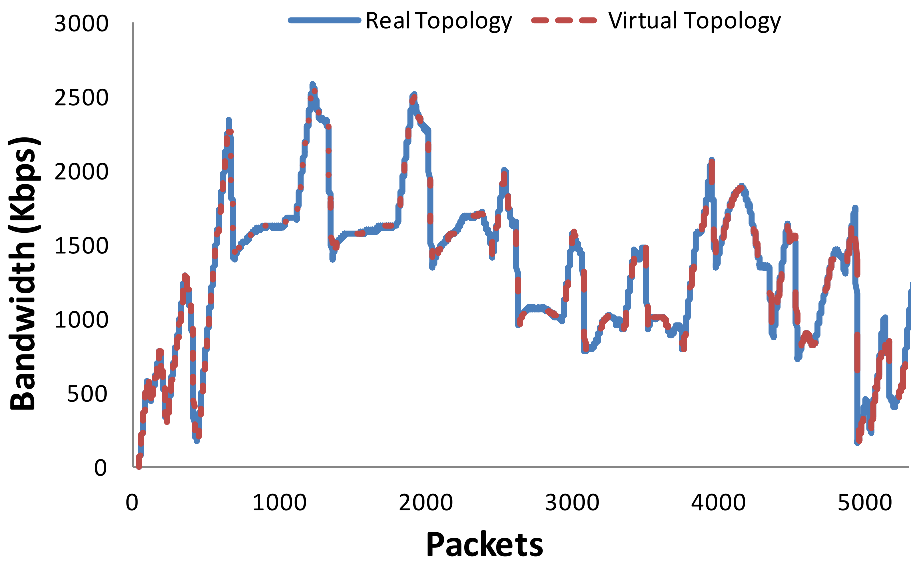


Figure 11. Bandwidth in topology 2, link at 10 Mbps

Figure 12 shows the comparison between the values of the bandwidth consumed in the real topology and in the virtual topology, when the link between switches has 100 Mbps. The data presents similar value for both topologies. The mean value of bandwidth in real topology is 1340.22 Kbps while for virtual topology is 1350.78 Kbps. The maximum value for both real and virtual topologies is the same 2592.67 Kbps. The minimum value has been slightly different, 162.72 for real topology and 173.57 for virtual topology (for the minimum, the initial 20 packets are not taken into account because the streaming was initiating). The variance of the data is 249230.523 for real topology and 248179.04 for virtual topology. The standard deviation is 499.22 in real topology and 498.17 in virtual topology. The data does not follow a normal distribution neither in real nor in virtual topology.

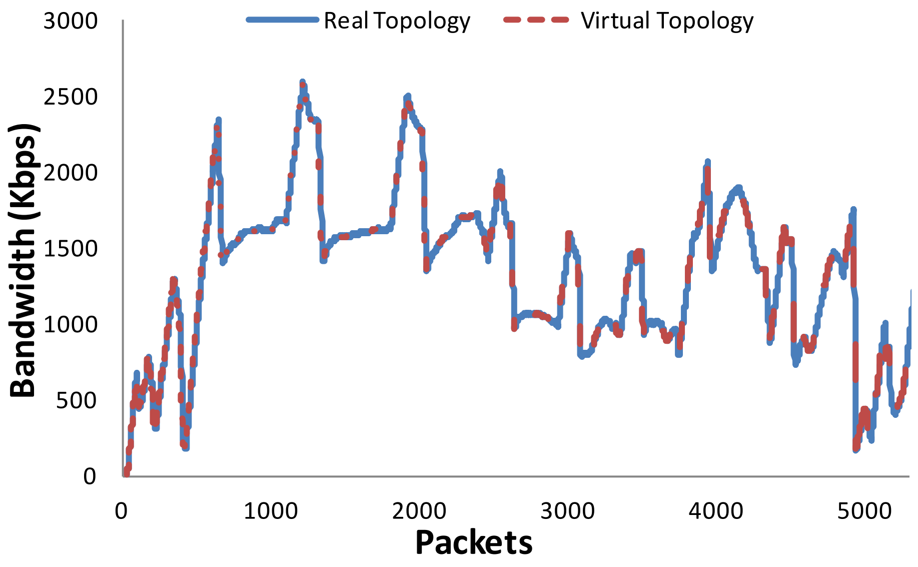


Figure 12 Bandwidth in topology 2, link at 100 Mbps

4.1.3 Bandwidth consumed in topology 3

When we study delay in topology 3, we can observe three different cases, when the WAN link between routers is established at 2, 4 and 8 Mbps.

Figure 13 shows the consumed bandwidth values of the real topology and the virtual topology, when the WAN link between routers is configured at 2 Mbps. Both topologies have similar data. The mean consumed bandwidth in real topology is 1335.49 Kbps while for virtual topology is 1372.51 Kbps. The maximum value for real topology is 1963.49 Kbps, while for virtual topology is 1996.03. The minimum value has been the same in both cases, 173.57 Kbps (for the minimum, the initial 20 packets are not taken into account because the streaming was initiating). The variance of the data is 218658.95 for real topology and 225180.89 for virtual topology. The standard deviation is 467.60 in real topology and 474.53 in virtual topology. The data does not follow a normal distribution neither in real nor virtual topology.

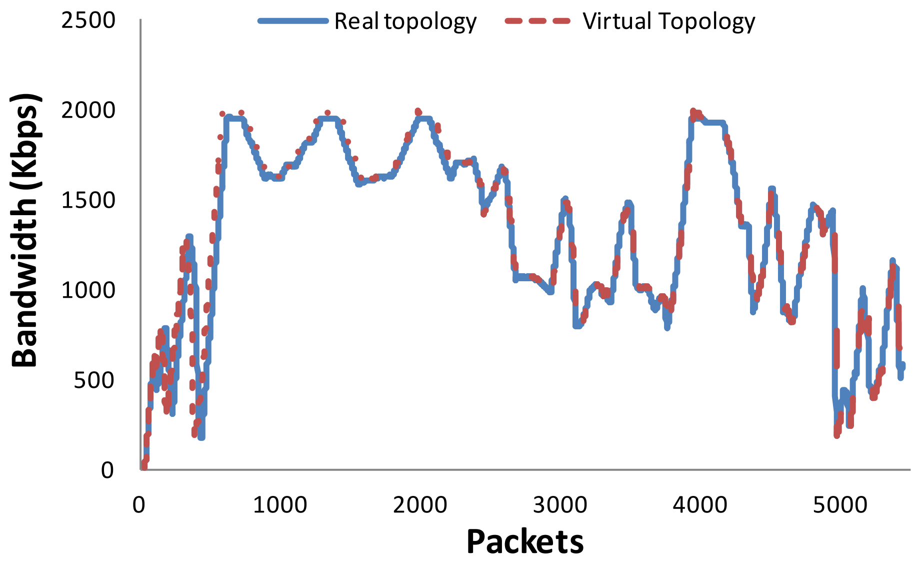


Figure 13 Bandwidth in topology 3, WAN link at 2 Mbps

In Figure 14 we can observe the consumed bandwidth for the real topology and the virtual topology, when the WAN link between routers is configured at 4 Mbps. The data presents similar value in both topologies. The mean value of bandwidth in the real topology is 1343.47 Kbps while for the virtual topology is 1353.89 Kbps. The minimum and maximum values for real topology and virtual topology are the same, 162.72 Kbps and 2505.89 Kbps respectively (for the minimum, the initial 20 packets are not taken into account because the streaming was initiating). The variance of the data is 245943.93 for real topology and 244590.52 for virtual topology. The standard deviation is 495.92 in real topology and 494.56 in virtual topology. The data does not follow a normal distribution neither in real nor virtual topology.

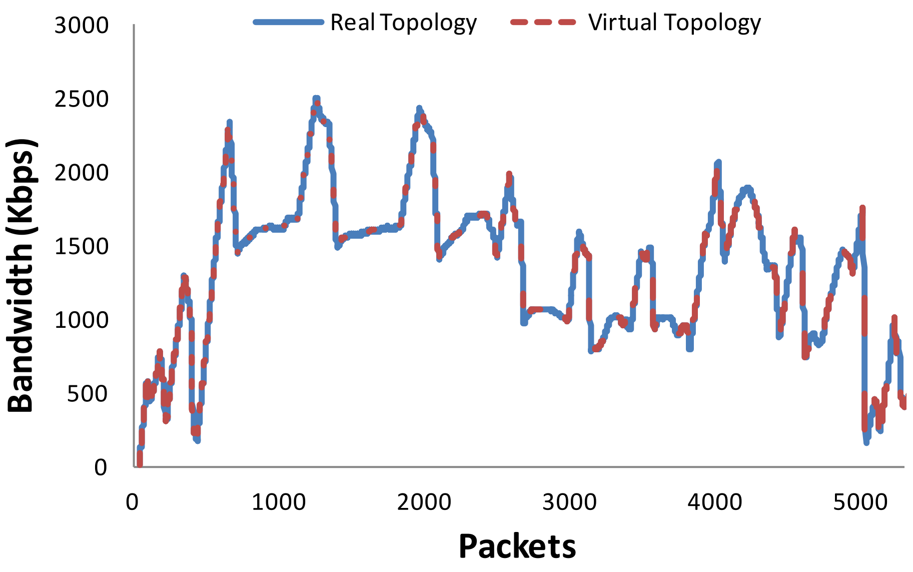


Figure 14 Bandwidth in topology 3, WAN link at 4 Mbps

Figure 15 shows the values of the bandwidth consumed by the real topology compared with the values of virtual topology, when the WAN link between routers is consumed at 8 Mbps. The data presents similar value for both topologies. The mean value of bandwidth in real topology is 1341.80 Kbps while for virtual topology is 1352.85 Kbps. The minimum and maximum values for real topology and virtual topology are the same, 162.72 and 2592.67 Kbps respectively (for the minimum, the initial 20 packets are not taken into account because the streaming was initiating). The variance of the data is 248882.63 for real topology and 247659.12 for virtual topology. The standard deviation is 498.88 in real topology and 497.65 in virtual topology. The data does not follow a normal distribution neither in real nor virtual topology.

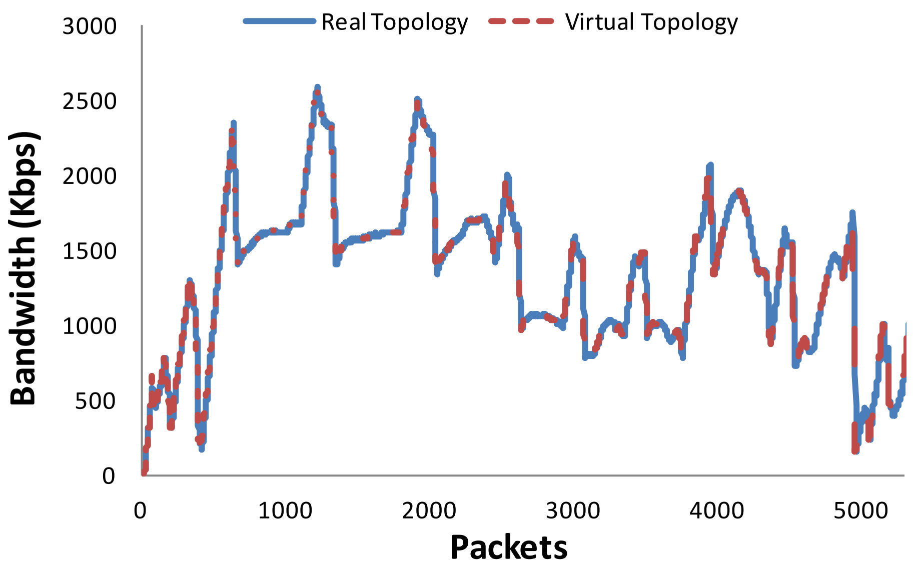


Figure 15 Bandwidth in topology 3, WAN link at 8 Mbps

4.2 Delay

In this subsection we present the results related to the delay obtained for the three topologies.

4.2.1 Delay in topology 1

Figure 16 shows the delay of the real topology and the delay of the virtual topology. The mean value of the delay in the real topology is 9.34 ms while for virtual topology is 9.32 ms. The minimum values are 0 ms and 0.02 ms for real topology and virtual topology respectively. The maximum values are 94.97 and 93.39 ms for real topology and virtual topology respectively. The variance of the data is 75.36 for real topology and 71.36 for virtual topology. The standard deviation is 8.68 in real topology and 8.45 in virtual topology. The percentage of difference between standard deviations is higher than in the bandwidth cases. The data does not follow a normal distribution neither in real or virtual topology.

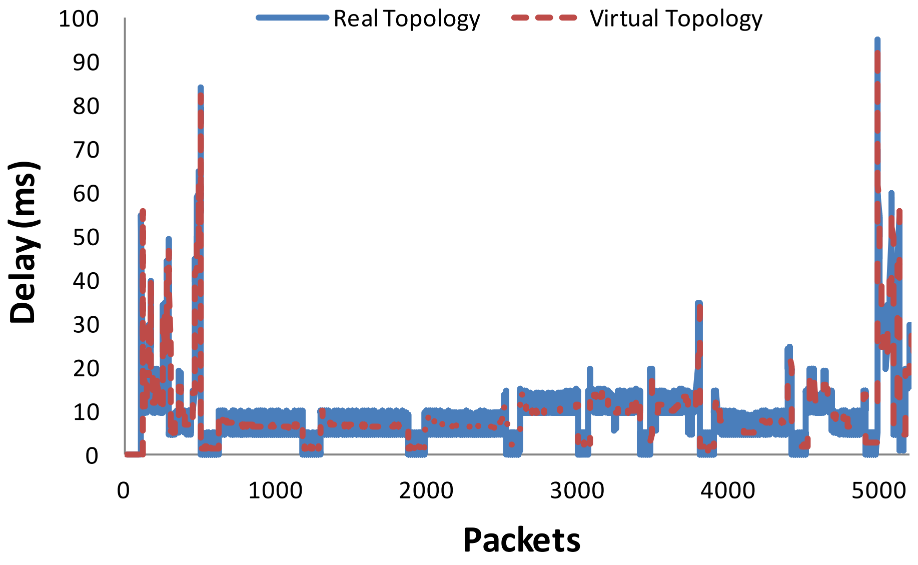


Figure 16. Delay in topology 1, link at 1 Gbps.

4.2.2 Delay in topology 2

When we study the delay in topology 2, we can observe two different cases, when the link between switches is established at 10 or 100 Mbps.

In Figure 17 we can see the values of the delay of both the real topology and the virtual topology. The data presents similar value for both topologies. The mean value of the delay in the real topology is 9.51 ms while for the virtual topology is 9.39 ms. The minimum and maximum values are 1.11 ms and 94.99 ms for real topology and 1.6 ms and 93.31 ms for virtual topology respectively. The variance of the data is 74.315 for the real topology and 69.427 for the virtual topology. The standard deviation is 8.62 in the real topology and 8.33 in the virtual topology. The data does not follow a normal distribution neither in real or virtual topology.

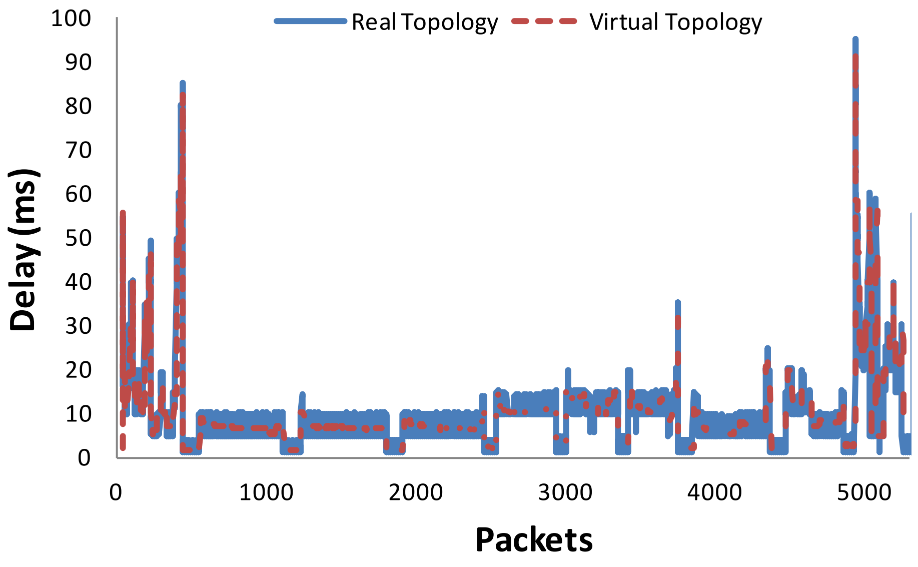


Figure 17. Delay in topology 2, link at 10 Mbps.

Figure 18 shows the delay values for both the real topology and the virtual topology. The data presents similar value for both topologies. The mean value of the delay in the real topology is 9.49 ms while for the virtual topology is 9.38 ms. The minimum and maximum values are for the real topology are 0 ms and 95.04 ms, respectively, and 0.96 ms and 93.31 ms for the virtual topology respectively. The variance of the data is 75.24 for real topology and 69.31 for the virtual topology. The standard deviation is 8.67 in the real topology and 8.33 in virtual topology. The data does not follow a normal distribution neither in real or virtual topology.

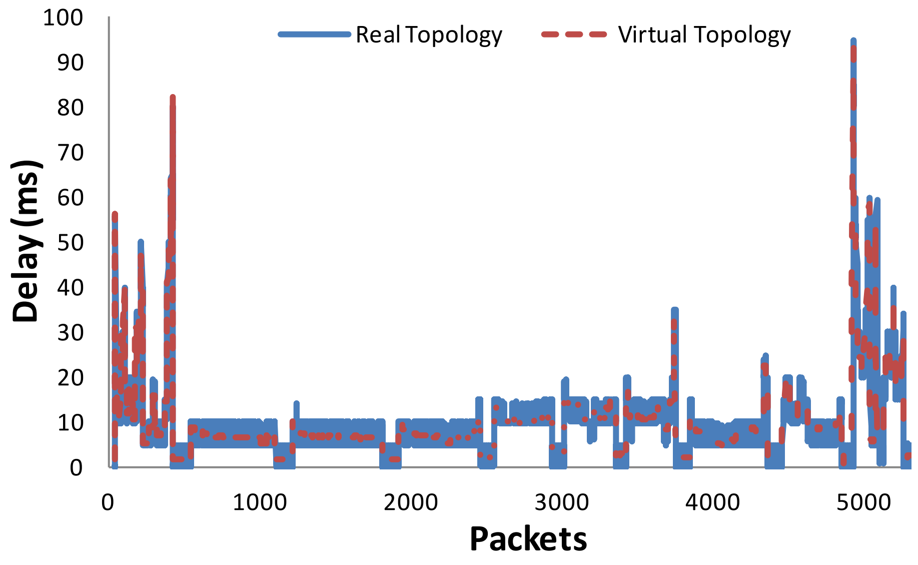


Figure 18 Delay in topology 2, link at 100 Mbps.

4.2.3 Delay topology 3

As in the previous subsection, when we study delay in topology 3, we can observe three different cases, when the WAN link between routers is configured at 2, 4 and 8 Mbps.

In Figure 19, it is observed the data gathered about the delay for both topologies, when the WAN link between routers is configured at 2 Mbps. The highest peaks are similar in both cases, but the lowest peaks only appear in the virtual topology. The mean value of delay for the real topology is 9.56 ms, similar to the mean value for the virtual topology, 9.39 ms. The highest delay in real topology is 85.15 ms, and the obtained value for the virtual topology is 82.38 ms. The minimum delay is 5.41ms for the real topology and only 1.69 for the virtual topology. The variance is 63.20 for the real topology and 61.19 for the virtual topology. Regarding to the standard deviation, it is obtained 7.95 for the real topology and 7.82 for the virtual topology. Finally the distribution of both sets of data follows a non-normal distribution.

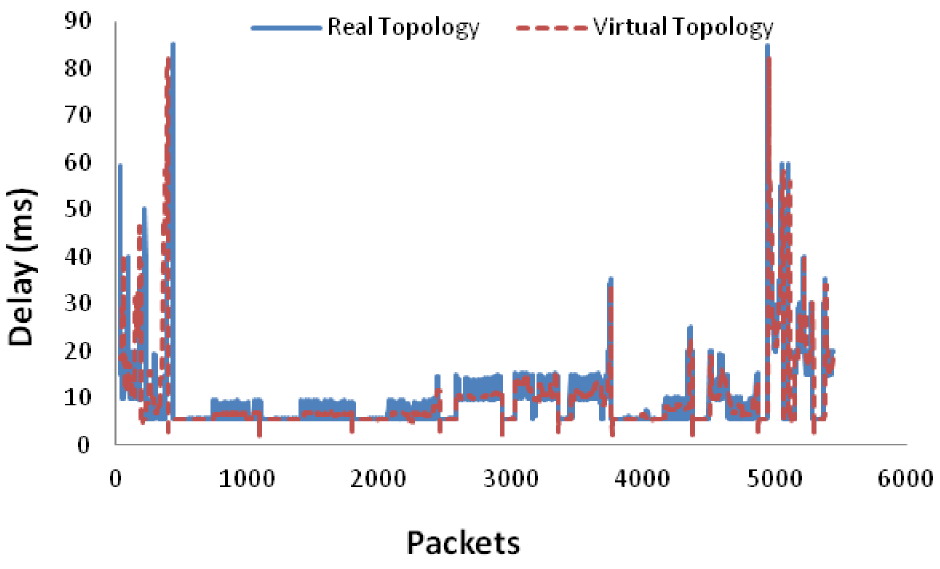


Figure 19. Delay in topology 3, WAN link at 2 Mbps.

The delay of the real and virtual topologies, when the WAN link between routers is configured at 4 Mbps, is shown in Figure 20. The peaks are placed in similar position for both topologies, but the real topology presents values in the stable periods. The mean for the real topology is 9.45 ms, quite close to the mean of the virtual topology, which is 9.37ms. The minimum delay is 1.69 ms for the virtual topology and 2.57 ms for the real topology. Regarding to the maximum value, the real topology provides higher value, 95.09 ms (the virtual topology has 93.37ms). The variance is higher for the real topology, 73.45, than for virtual topology, 67.55. The standard deviation is 8.57 for real topology and 8.22 for virtual topology. Finally the data of both topologies follows a non-normal distribution.

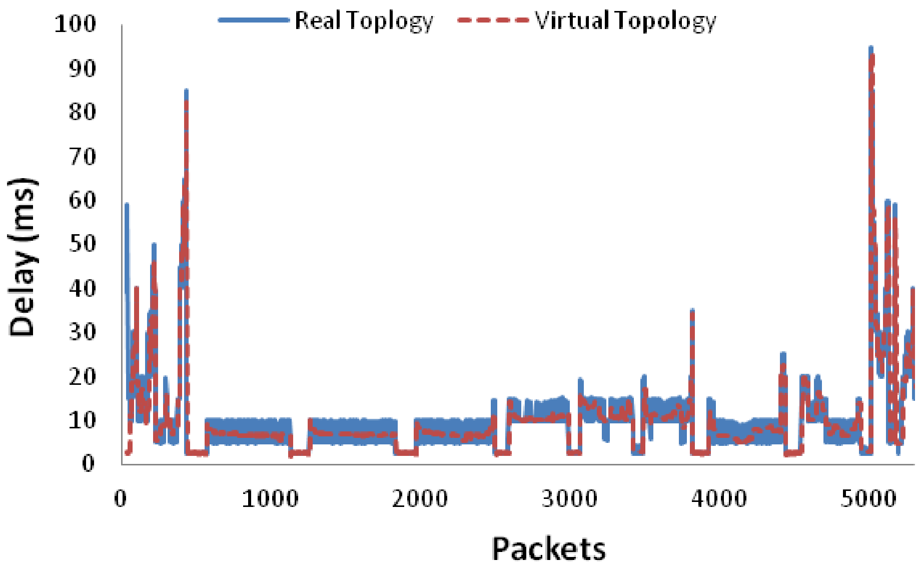


Figure 20. Delay in topology 3, WAN link at 4 Mbps.

In Figure 21, it is presented the data about the delay for the virtual and real topologies, when the WAN link between routers is configured at 8 Mbps. The mean value of delay for the real topology is 9.45 ms, similar to the mean value for the virtual topology, 9.41 ms. The highest delay in real simulations is 95.06 ms, which is higher than the delay of the virtual topology (93.43 ms). The minimum delay is 1.20 ms for the real topology, which is lower than the minimum delay for the virtual topology (1.34 ms). Regarding to the variance, it is higher for the real topology, 75.56, than for the virtual topology, 68.46. The standard deviation for the real topology is 8.63 and for the virtual topology is 8.27. Finally the distribution of both sets of data follows a non-normal distribution.

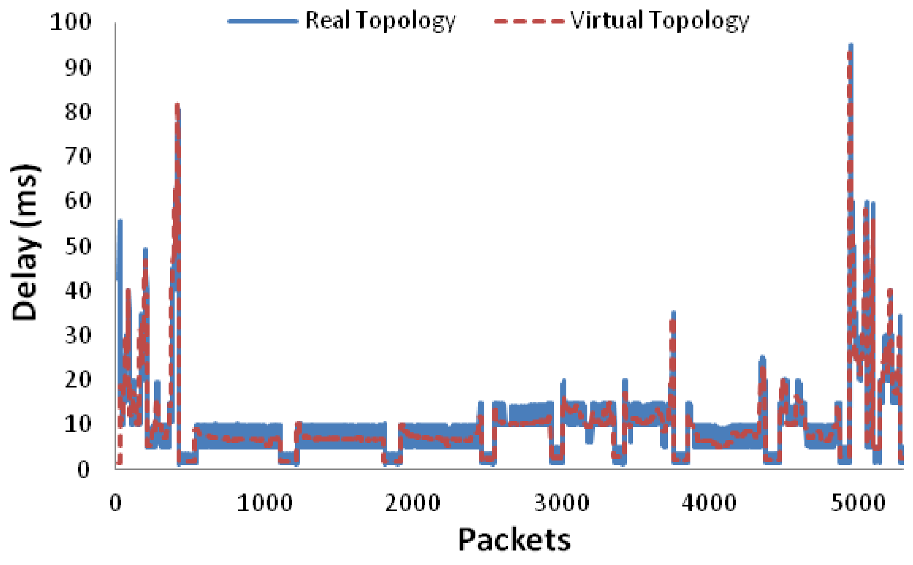


Figure 21. Delay in topology 3, WAN link at 8 Mbps.

4.3 Jitter

In this section we present the results related the delay obtained for the three topologies.

4.3.1 Jitter in topology 1

Figure 22 shows the values obtained for the jitter in the real topology and in the virtual topology. The values of the real topology are higher than in the virtual topology. The mean value of jitter in the real topology is 0.044 ms, while for virtual topology is lower than 0.004. The minimum and maximum values are the same for both topologies 0 and 3.26. The variance is <0.01 for both topologies. The standard deviation is 0.065 in the real topology and 0.056 in the virtual topology. Respecting to the distribution, the data for both topologies follow a non-normal distribution.

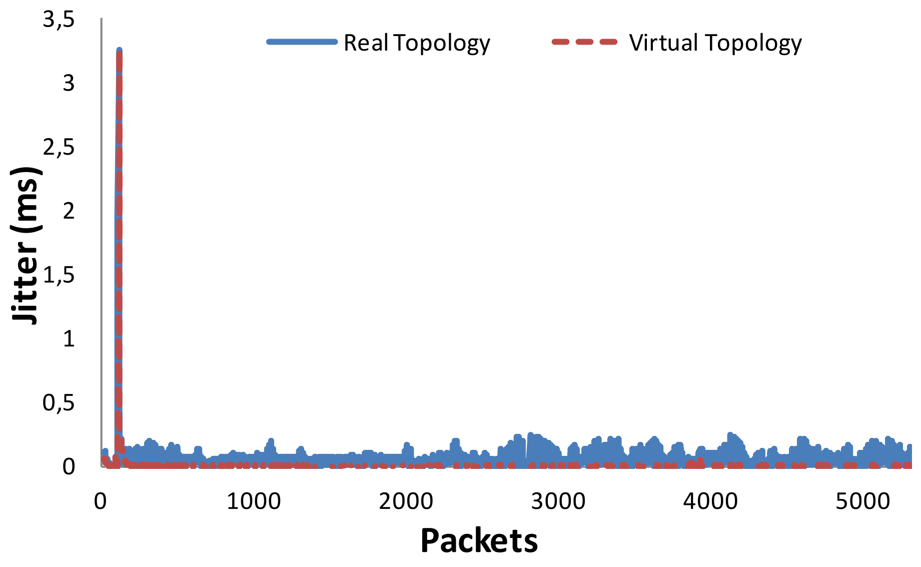


Figure 22. Jitter in topology 1, link at 1 Gbps.

4.3.2 Jitter in topology 2

When we study the jitter in topology 2, we can observe two different cases, when the link between switches is set up at 10 or 100 Mbps.

Figure 23 shows the values of the jitter in the real topology and in the virtual topology. The values of the real topology are higher than of the virtual topology. The mean value of jitter for the real topology is 0.040 ms, while for the virtual topology is 0.004. The maximum value of the jitter is 1.26 ms for the real topology, while for the virtual topology is 0.84 ms. The minimum value is 0 in both cases. The variance is <0.01 for both topologies. The standard deviation is 0.041 in the real topology and 0.034 in the virtual topology. The data for both topologies follow a non-normal distribution.

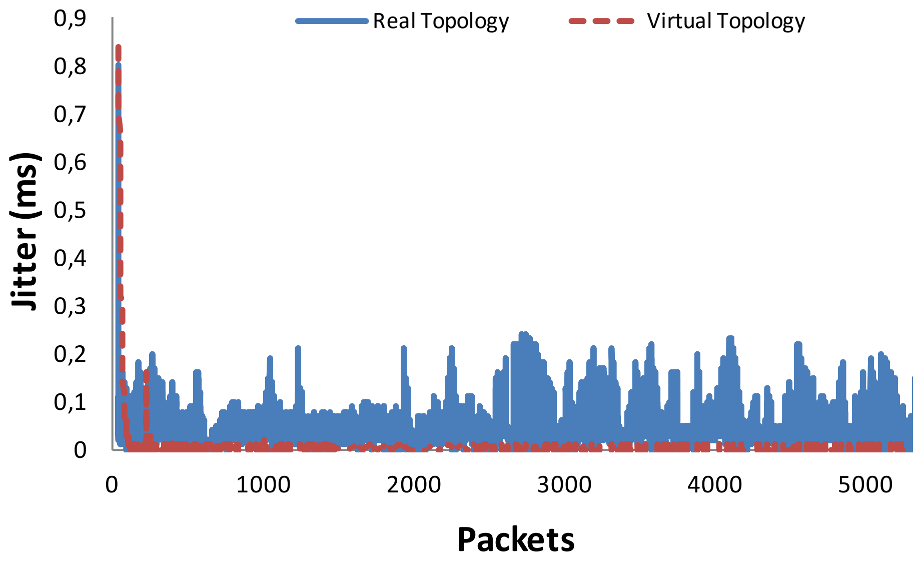


Figure 23. Jitter in topology 2, link at 10 Mbps.

Figure 24 shows the values of the jitter in real topology and in the virtual topology. The values of the real topology are higher than of the virtual topology. The mean value of jitter for the real topology is 0.044 ms, while for virtual topology is 0.012 ms. The minimum and maximum values are the same for both topologies 0 ms and 3.33 ms respectively. The variance of the data is 0.0004 for the real topology and 0.0176 for the virtual topology. The standard deviation is 0.07 in the real topology and 0.13 in the virtual topology. The data for both topologies follow a non-normal distribution.

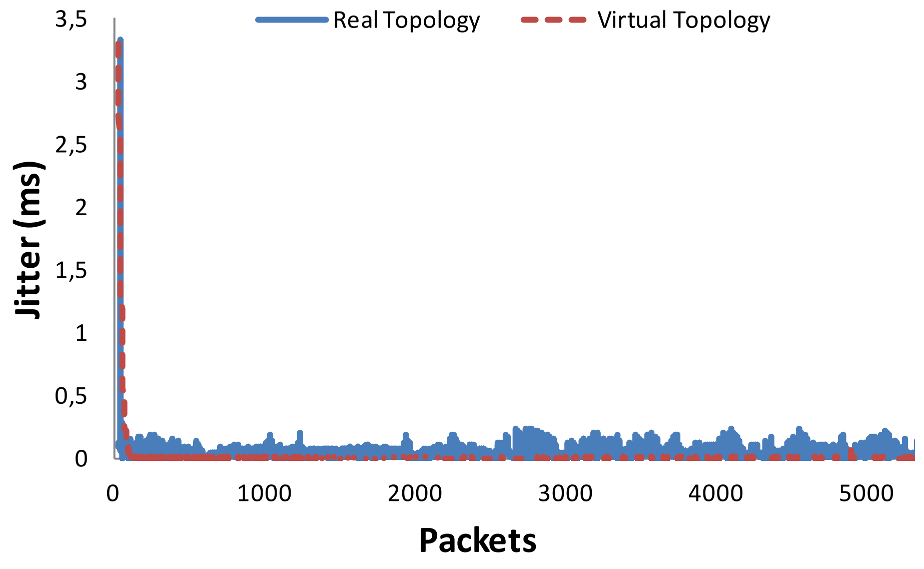


Figure 24. Jitter in topology 2, link at 100 Mbps.

4.3.3 Jitter in topology 3

As in previous subsections, when we study the jitter in topology 3, we can observe three different cases, when the WAN link between routers is configured at 2, 4 and 8 Mbps.

In Figure 25 we can see the values of the jitter of the real topology and of the virtual topology, when WAN link between routers is configured at 2 Mbps. The data of the real topology presents more peaks than the data of the virtual topology. The mean value of the jitter in real topology is 0.05 ms while the mean value of the virtual topology is 0.03 ms. The minimum and maximum values for the real topology are 0 and 5.25ms, while for the virtual topology are 0 and 5.32 ms. The variance of the data is 0.14 for the real topology and 0.13 for the virtual topology. The standard deviation is 0.14 for the real topology and 0.12 for the virtual topology. The data does not follow a normal distribution neither in real nor virtual topology.

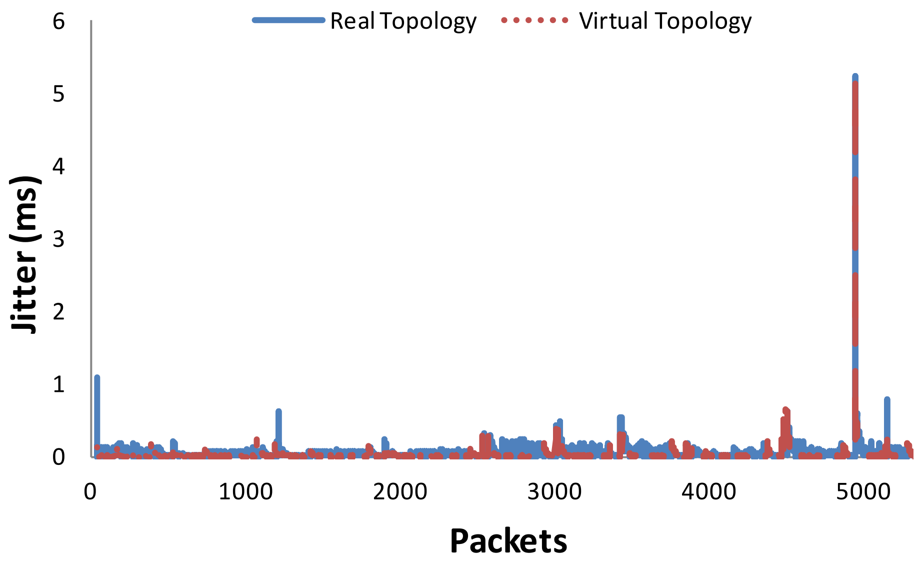


Figure 25 Topology 3, WAN link at 2 Mbps

We can see the values of the jitter, of the real topology and of the virtual topology, in Figure 26, when the WAN link between routers is configured at 4 Mbps. The jitter of the real topology presents higher mean values than in the virtual topology (0.4 ms for the real topology and 0.3 ms for the virtual topology). The minimum and maximum values are 0 and 0.67 ms for the real topology and 0 and 0.76 ms for virtual topology respectively. Even having higher mean values for the real topology, the maximum value (highest peak) is higher for the virtual topology. Regarding to the standard deviation, the values are almost the same, 0.5 and 0.6 for the real and the virtual topology respectively. The variance of the data is <0.01 in both cases. Finally, for the distribution of the data, we can affirm that the data do not follow a normal distribution.

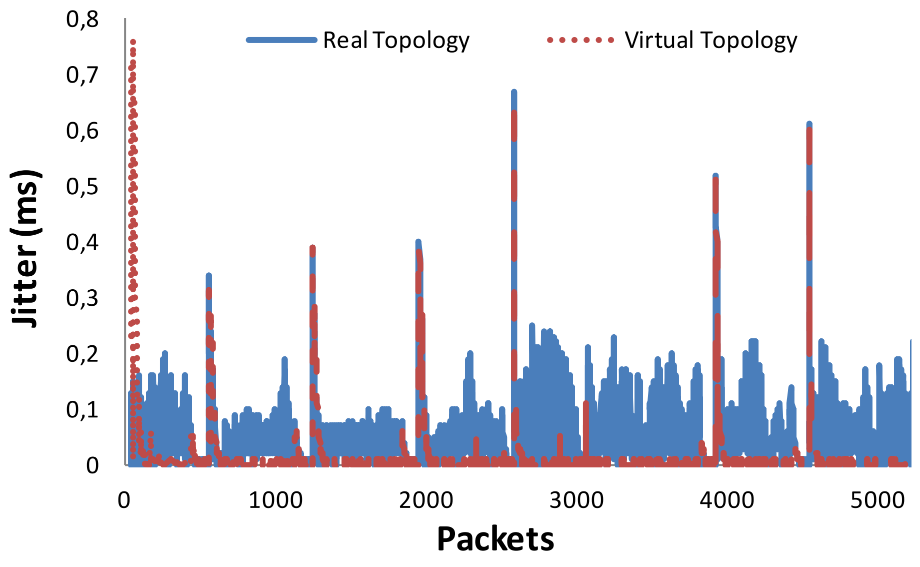


Figure 26. Topology 3, WAN link at 4 Mbps.

Figure 27 shows the values of the jitter in real topology and in virtual topology, when the WAN link between routers is configured at 8 Mbps. Values from real topology are higher than in virtual topology. The mean value of the jitter for the real topology is 0.04ms, while for the virtual topology is lower than 0.01. The maximum value of the jitter is 0.24 ms for the real topology, while for the virtual topology is 0.13 ms, almost half than in real topology. The minimum value is 0 in both cases. The variance is <0.01 for both topologies. As it is expected, the value of the standard deviation is much higher in the real topology, 0.04, than in the virtual topology, which is lower than 0.01. Respect to the distribution, the data for both topologies follow a non-normal distribution.

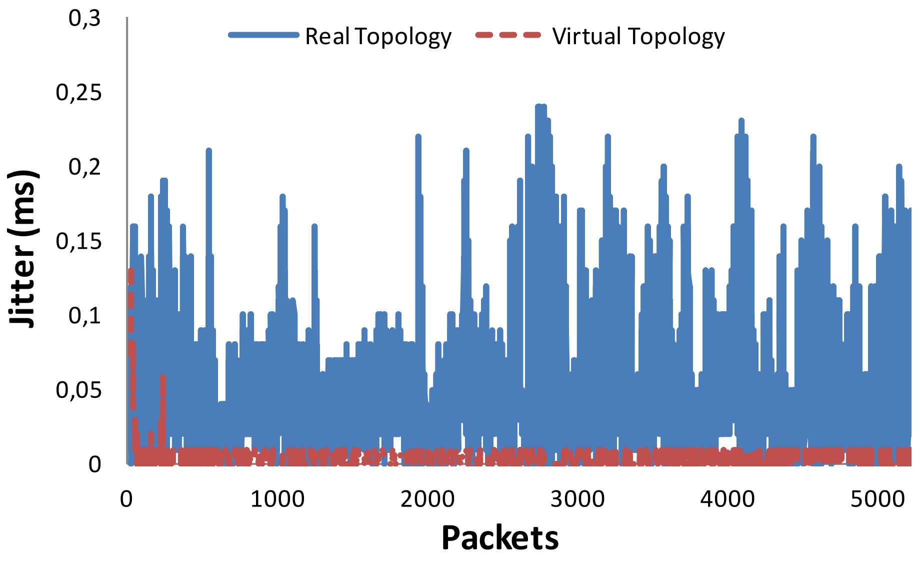


Figure 27 Topology 3, WAN link at 8 Mbps

5. Conclusion

This paper has studied the performance of different cases in both real topology and virtual topology, when multimedia is being delivered through the topology. The studied parameters have been the consumed bandwidth (throughput), delay and jitter. The study shows how realistic are the emulator Mininet when the performance of multimedia streaming is measured.

We have obtained different minimum value in direct connection than in the connection through switches (using 10Mbps and 100Mbps) in the virtual topology. So, the fact of introducing network devices in the virtual topology affects to the consumed bandwidth, while the real topology does not affect to them. To change from 10Mbps to 100Mbps does not affect to the data in both cases. For routers cases, with WAN link between them, at 2 Mb, 4 MB and 8 MB, we obtained the same results in real topology than in the virtual topology.

In the delay study the real topology has lower minimum and higher maximum than the virtual topology in all cases.

The mean and the maximum peak values of the jitter in the real topology are higher than in the virtual topology when the throughput is high, but when the throughput is low, the maximum peak is higher in the virtual topology.

This study can be applied to many other study cases, such as when using routing protocols [35]. In our future work we are going to compare it with Openflow and in more complex topologies.

Acknowledgement

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] Internet Engineering Task Force (IETF). Available at: http://www.ietf.org/ [Last access: December 28, 2015]
[2] What’s Software-Defined Networking (SDN).(Online article). Available at sdx central website: https://www.sdxcentral.com/resources/sdn/what-the-definition-of-software-defined-networking-sdnYoung, G. O., “Synthetic structure of industrial plastics”, in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64. [Last access: December 28, 2015]
[3] Software-Defined Networking (SDN) Definition).(Online article). Available athttps://www.opennetworking.org/sdn-resources/sdn-definition [Last access: December 28,

2015]
[4] Software-Defined Networking: A Perspective from within a Service Provider Environment. Available at: https://tools.ietf.org/pdf/rfc7149.pdf [Last access December 28, 2015]
[5] What is OpenFlow?. Available at: http://archive.openflow.org/wp/learnmore/ [Last access December 28, 2015]
[6] Software-Defined Networking: The New Norm for Networks. Available at Open Network Foundation website: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf [Last access December 28, 2015]
[7] NFV standard. Available at IETF website: https://portal.etsi.org/tb.aspx?tbid=789&SubTB=789,795,796,801,800,798,799,797,802 [Last access December 28, 2015]
[8] Perrin, S., Hubbard, S., "Practical implementation of SDN & NFV in the WAN" White paper, Heavy Reading, October 2013, https://networkbuilders.intel.com/docs/HR-Intel-SDN-WP.pdf
[9] RTP: A Transport Protocol for Real-Time Applications. Available at IETF website: https://www.ietf.org/rfc/rfc1889.txt [Last access December 28, 2015]
[10] An Instant Virtual Network on your Laptop (or other PC). Available at Mininet website: http://mininet.org/ [Last access December 28, 2015]
[11] Hu, F., Hao, Q., Bao, K.,"A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation", Communications Surveys & Tutorials, IEEE (Volume:16, Issue:4), 22 may 2014, pp. 2181 – 2206. http://dx.doi.org/10.1109/COMST.2014.2326417
[12] Scott-Hayward, S., O'Callaghan, G., Sezer, S., "SDN Security: A Survey", IEEE SDN for Future Networks and Services (SDN4FNS 2013), 11-13 Nov. 2013, Trento, Italy, pp. 1-7. http://dx.doi.org/10.1109/SDN4FNS.2013.6702553
[13] Kreutz, D., Ramos, F.M.V., Esteves Verissimo, P., Esteve Rothenberg, C., Azodolmolky, S., Uhlig, S., "Software-Defined Networking: A Comprehensive Survey", Proceedings of the IEEE, Volume 103, Issue 1, Jan. 2015, pp. 14-76. http://dx.doi.org/10.1109/JPROC.2014.2371999
[14] Kaur, K., Singh, J., Ghumman, N. S., “Mininet as Software Defined Networking Testing Platform”, International Conference on Communication, Computing & Systems (ICCCS. 2014), 20 Feb - 21 Feb 2014, Chennai, India
[15] Paasch, C., Ferlin, S., Alay, O., Bonaventure, O., “Experimental Evaluation of Multipath TCP Schedulers”, ACM SIGCOMM Capacity Sharing Workshop (CSWS), 2014. ACM., August 18, 2014, Chicago, IL, USA. http://dx.doi.org/10.1145/2630088.2631977
[16] de Oliveira, R.L.S., Shinoda, A.A., Schweitzer, C.M., Rodrigues, P. L., "Using Mininet for Emulation and Prototyping Software-Defined Networks", IEEE Communications and Computing (COLCOM 2014), 4-6 June 2014, Bogota, Colombia, pp. 1-6. http://dx.doi.org/10.1109/ColComCon.2014.6860404
[17] Keti, F., Askar, S., "Emulation of Software Defined Networks Using Mininet in Different Simulation Environments", 6th International Conference on. IEEE Intelligent Systems, Modelling and Simulation (ISMS 2015), Kuala Lumpur, Malaysia, 2015. p. 205-210.

http://dx.doi.org/10.1109/ISMS.2015.46

[18] Azizi, M., Benaini, R., Ben Mamoun M., "Delay Measurement in Openflow-Enabled MPLS-TP Network", Modern Applied Science, Vol 9, No 3, 2015, Canadian Center of Science and Education, January 10, 2015. http://dx.doi.org/10.5539/mas.v9n3p90

[19] Gupta, M., Sommers, J., Barford, P. "Fast, accurate simulation for SDN prototyping", 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, 12-15 August, 2013, Hong Kong. http://dx.doi.org/10.1145/2491185.2491202

[20] Panwaree, P., Kim, J., Aswakul, C., "Packet Delay and Loss Performance of Streaming Video over Emulated and Real OpenFlow Networks.", The 29th International Technical Conference on Circuit/Systems Computers and Communications (ITC-CSCC), Phuket, Thailand, July 1-4, 2014

[21] Megyesi, P., Botta, A., Aceto, G., Pescapè, A., Molnár, S., "Available Bandwidth Measurement in Software Defined Networks.", SAC 2016,April 04 - 08, 2016, Pisa, Italy. http://dx.doi.org/10.1145/2851613.2851727

[22] Noghani, K. A., M. Oguz Sunay, M., “Streaming Multicast Video over Software-Defined Networks”, IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems (MASS 2014), 28-30 Oct. 2014, Philadelphia, PA, USA, pp. 551 – 556. http://dx.doi.org/10.1109/MASS.2014.125

[23] Jarschel, M., Wamser, F., Hohn, T., Zinner, T., Tran-Gia, P., “SDN-based Application-Aware Networking on the Example of YouTube Video Streaming”, 2013 Second European Workshop on Software Defined Networks (EWSDN 2013), Oct. 10 - 11, 2013, Berlin, Germany, pp: 87-92. http://doi.ieeecomputersociety.org/10.1109/EWSDN.2013.21

[24] Salsano, S., Ventre, P. L., Prete, L., Siracusano, G., Gerola, M., Salvadori, E., "OSHI-Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds).", 2014 Third European Workshop on Software Defined Networks (EWSDN), IEEE, 1-3 September 2014, Budapest, Hungary, p. 13-18. http://dx.doi.org/10.1109/EWSDN.2014.38

[25] Lantz, B., Heller, B., McKeown, N. A network in a laptop: rapid prototyping for software-defined networks. 9th ACM SIGCOMM Workshop on Hot Topics in Networks, ACM, 20 – 21 October, 2010, Monterey, CA, USA, Article No. 19. http://dx.doi.org/10.1145/1868447.1868466

[26] CISCO. “Support – Cisco 2800 Series Integrated Services Routers (Cisco 2811 Integrated Services Router)”. Available at: http://www.cisco.com/c/en/us/support/routers/2811-integrated -services-router-isr/model.html [Last access December 28, 2015]

[27] CISCO. “Support – Cisco Catalyst 3560 Series Switches (Cisco Catalyst 3560-24PS Switch)”. Available at: http://www.cisco.com/c/en/us/support/switches/catalyst-3560-24ps-switch/model. html [Last access December 28, 2015]

[28] Videolan software. Available at Videolan website: http://www.videolan.org/vlc/ [Last access December 28, 2015]

[29] Wireshark software. Available at Wireshark website: https://www.wireshark.org/ [Last access December 28, 2015]

[30] NS-3 (Online article), Available at NSNAM website: https://www.nsnam.org/overview/key-technologies/ [Last access December 28, 2015]

[31] EstiNet Technologies Inc. Available at EstiNet Technologies website: http://www.estinet.com/index.php [Last access December 28, 2015]

[32] The OpenDaylight Platform. (Online article), Available at The OpenDaylight website: https://www.opendaylight.org/ [Last access December 28, 2015]

[33] Component-based software defined networking framework. (Online article) Available at Ryu SDN Framework community website: http://osrg.github.io/ryu/ [Last access December 28, 2015]

[34] Project Floodlight. (Online article) Available at Project Floodlight website: http://www.projectfloodlight.org/floodlight/ [Last access: December 28, 2015]

[35] S. Sendra, P.A. Fernández, M.A. Quilez, J. Lloret, Study and performance of interior gateway IP routing protocols, Network Protocols and Algorithms 2 (4), 88-117. 2011. http://dx.doi.org/10.5296/npa.v2i4.547