On the Use of Emulation Techniques over the ns-3 Platform to Analyze Multimedia Service QoS

Juan Arroyo, Luis Diez(✉), and Ramón Agüero

University of Cantabria, Santander, Spain
jaarsolares@gmail.com, {ldiez,ramon}@tlmat.unican.es

Abstract. This paper presents an assessment study on the use of emulation techniques to evaluate the performance of multimedia services, in particular a video streaming application. The proposed methodology exploits the emulation functionality that has been integrated into the ns-3 simulation platform, as well as virtualization techniques. We describe the requirements and the main limitations of the scheme, in order to offer an appropriate accuracy. Last, and in order to assess the feasibility of the proposed scheme and its potential, we carry out a study of various QoS parameters of a video streaming application.

1 Introduction

The growing presence of advanced mobile terminals and the way they are being used are causing a great change on the mobile data traffic. According to the latest reports [1], video streaming sessions are believed to become the predominant type of traffic in the forthcoming years. This is of course posing several challenges to the mobile operators, that need to face an unexpected resource demand. Different algorithms, techniques and solutions, are being proposed to overcome this situation.

In order to assess the feasibility and efficiency of such proposals, the use of real implementations is usually not an option, since the available technology might not be able to provide the required functionality or simply because it is not realistic trying to emulate the conditions under which the assessment should be carried out. As a consequence, the use of simulation techniques is gaining more and more popularity. One of the criticisms about this approach is that in some cases, this simulation-based analysis is not realistic, especially considering the characteristics of the lower layers as well as the details of the real services.

In this paper we focus on this latter point, since we study an approach to enhance the realism of a simulation-based study, thanks to the recent advances in virtualization. The Network Simulator 3 (ns-3) framework [2] allows the use of virtual nodes, able to run real applications, which can be connected to the simulator so that they would use the network technology it provides. That would bring several advantages, since we could, for instance, better understand the effect that a particular type of errors might have over a multimedia-flow; it goes

R. Agüero et al. (Eds.): MONAMI 2015, LNICST 158, pp. 196–208, 2015.
DOI: 10.1007/978-3-319-26925-2_15

without saying that being able to reproduce similar conditions is one of the clear limitations of carrying out a characterization over real testbeds.

The rest of the paper is structured as follows: Sect. 2 discusses some works that have used the same simulation strategy to analyze the performance of some services. Afterwards, Sect. 3 discusses the main conclusions that can be extracted when exploiting the ns-3 as an emulation tool; first, we analyze the constraints that are inherent to ns-3, and afterwards, the tool is used to study the behavior of video services over multi-hop wireless networks. Finally, Sect. 4 concludes the paper, advocating a number of research lines that we will tackle in our future work, exploiting the acquired know-how and the possibilities that are brought about by the combination of a network simulation tool and virtualization techniques.

2 Foundations and Related Work

Network simulation platforms have become a key tool for the scientific and research community. In spite of the recent looming of various testbed initiatives, simulation still offers a greater degree of flexibility, making it a very attractive tool for the initial steps of the research of novel aspects and topics (for instance, network coding). Among others, one of its main advantages is that it brings about the possibility to analyze large and complex network topologies, on both a cost and time effective manner. As a consequence, there are still a large number of research works that involve the use of network simulation.

In any case, and despite the aforementioned advantages, the interaction with the *real world* becomes in some cases an essential aspect. It is particularly relevant when the goal is to analyze the impact of specific networking technology, entity or solution might have over specific services (e.g. real-time applications). In order to address this need, ns-3 provides a set of capabilities that allow its closer interaction with real entities at different levels [3], as it is depicted in Table 1. As can be seen, depending on the selected configuration, different entities of the communication system (final nodes or network infrastructure) can be simulated and afterwards integrated with real systems. In this work, we focus on the interaction of networks that are simulated in ns-3 with real nodes (indicated in Table 1 as **NS3-TAP**), exploiting virtualization capabilities; this configuration brings about the possibility of running real applications over simulated network infrastructures. In the remainder of the paper we will refer to this approach as *emulation*.

Table 1. NS3 interactions with *real world*

	Networks		
		Real	Simulated
Nodes	Real	Host	**NS3-TAP**
	Simulated	NS3-EMU	Simulation

In order to interact with real nodes, ns-3 provides an interface between the simulated and the real systems, the so-called *NS3-TAP*, which can be connected to virtual bridges of the host computer. The most typical setup for this emulation entails Linux Containers (LXC) [4], which offers the possibility of including *light* virtualized nodes, where real applications may be embedded. From the host point of view, a virtual bridge defines a network within the host, and both the container embedding the application and the *NS3-TAP* are connected to it. Besides, the *NS3-TAP* is also attached to a simulated node, and the data sent by the LXC virtual node can be in fact forwarded to the simulated node. Furthermore, the interaction with real applications requires that the simulated time (that internally managed by the simulator) is aligned with the system clock. To achieve this, ns-3 provides a configurable scheduling mechanism, which can be configured to use the system clock (instead of the internal simulated time) when scheduling the various events happening during a simulation. Throughout the paper we will study the performance and the limitations exhibited by the simulator when using this configuration, paying special attention to the accuracy of the real time scheduling.

There are some works that have successfully exploited this emulation capability over different network technologies and using various applications. For instance, in [5,6] the authors use the ns-3 emulation capability to study the performance of video applications over LTE networks, focusing on the impact that handovers might have over the Quality of Service (QoS) perceived at the receiver application. Nevertheless, they do not discuss whether the simulator has any limitations to carry out such type of experiments or how it was configured.

Other works have focused on the applications. For instance, [7] developed, over an emulated environment, a tool to measure both the QoS and the Quality of Experience (QoE) of video applications with a modified version of the VLC player. In this case, the authors do mention the real-time scheduling limitations (in particular the *HardLimit* parameter that will be extensively discussed in Sect. 3), but the results they discuss are rather limited, restricted to the particular scenario they used.

Finally, other works specifically study the limitations of the simulator, but they do it from a different perspective. The authors of [8] propose a distributed ns-3 emulation, to share the computational load between a number of real systems. Although this approach is able to reduce the computation time, there exist some network topologies that would not allow the use of distributed systems. One important example are wireless networks, where different nodes share the same physical channel, preventing the nodes from being moved to different machines, since that would not allow emulating the interaction of the lower layer mechanisms in an appropriate way.

3 Experimental Evaluation

In order to assess the feasibility of the ns-3 platform to exploit emulation techniques (i.e. run real applications/services over simulated network topologies), this section presents some of the most relevant results that were obtained after an extensive simulation campaign that had a twofold objective: first, Subsect. 3.1 discusses, based on a number of topologies with increasing complexity, the limitations exhibited by the simulator. Afterwards, Subsect. 3.2 discusses an evaluation of real video traffic (in terms of QoS) that was carried out as a proof-of-concept, describing the impact that this type of applications might have on the simulator.

It is important to mention that all the results presented herewith have been obtained with a laptop ASUS $S500CA$, equipped with an Intel Core $i5 - 3317U@1.7$ GHz microprocessor and 8 GB of RAM memory. We have used Ubuntu $14.04LTS(64bits)$ as the operating system.

3.1 Impact of the Simulator Performance on the Real-Time Scheduling

We start by studying the impact of the number of devices over the validity of the methodology. We are therefore interested in finding the limit that might guarantee an appropriate accuracy.

In order to better understand the results that will be shown afterwards, it is worth summarizing how ns-3 can be configured to use emulation techniques. ns-3 is an event-driven simulator, where the different events that happen during an experiment are enqueued according to the *simulation time* they take place. In its normal operation, as soon as one event has been handled, the following one is taken care of. However, emulation purposes the simulator exploits the system clock, and the events are processed according to the actual time. In this sense, if a large number of events take place within a short period, the simulator might get overloaded and a delay between the time an event had to be handled and the moment it actually was differs; we call this difference Simulation Delay (SD).

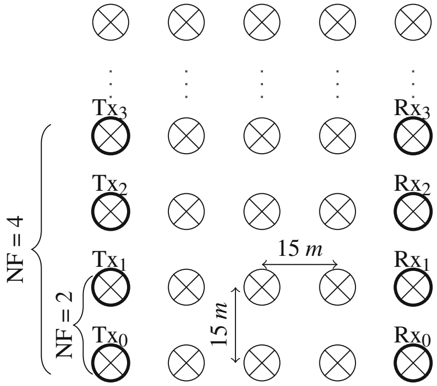


Fig. 1. Simulation scenario with a given number of traffic flows

The simulator allows the user to set a maximum value of SD by means of a configuration parameter called *HardLimit*; hence, if the SD exceeds this value, the simulation is aborted. In this section, we study the evolution of the SD under different network topologies. We pay special attention to the number of physical devices that are deployed, since they share the same physical channel, and this leads to a large number of events that are scheduled when a single packet is sent, regardless a node is directly involved in the communication or not. In this sense, all physical devices need to be checked when a packet is sent, which might lead to a bottleneck at running time. It is worth highlighting that this circumstance takes place at he physical layer, so that it could not be solved by using multiple machines, and it has also become an important challenge for the parallelization efforts that have been carried out within the simulator framework.

The topologies under study in this section consider a uniform grid structure, see Fig. 1. The nodes have been configured to use Ad hoc On-Demand Distance Vector Routing (AODV) as routing protocol, while the distance between the traffic sources and the corresponding destination nodes is 4 hops. The traffic flows are modeled as Constant Bit Rate (CBR) traffic and we have executed simulations of 15 s.

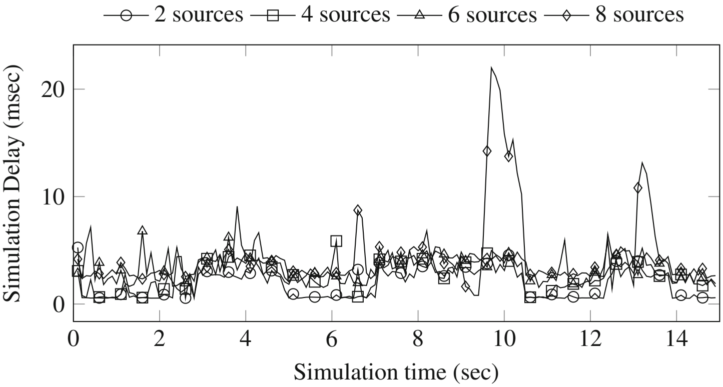
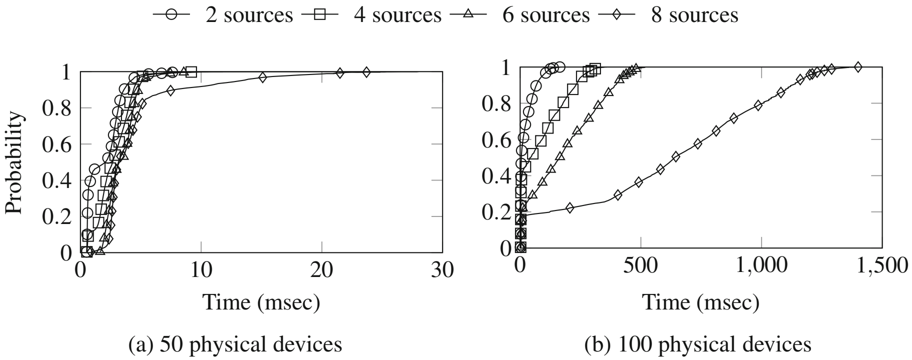


Fig. 2. Simulation delay evolution in a single experiment with 50 nodes

As an illustrative example, Fig. 2 shows the evolution of the SD value during one simulation, for different number of flows and considering 50 WiFi nodes. The results reveal that, in spite of the presence of delay peaks, the simulator is able to maintain a low and constant difference with the real time. This peaky effect at the beginning is due to the scheduling several of events at some particular moments (e.g. when the nodes send the AODV *HELLO* message). It can be therefore conclude that with an strict value for the *HardLimit* parameter some scenarios would be considered as unacceptable for certain requirements, while that particular threshold might be only exceeded at a particular time.

Figure 3 shows the Cumulative Distribution Function (CDF) of the SD value, initiating different number of flows in the scenario. The figures have been obtained by measuring the SD every 100 ms and for 20 simulations, yielding a number of around 3000 samples per configuration. As can be observed, ns-3 is able to appropriately simulate WiFi ad-hoc networks with up to 50 nodes with a reasonably good degree of accuracy. In this sense, Fig. 3a shows that more than 90 % of the samples showed a SD less than 5 ms, but for the scenario with the highest number of traffic sources, 8. In this latter case, the corresponding traffic jeopardizes the real-time accuracy.



(a) 50 physical devices

(b) 100 physical devices

Fig. 3. Simulation delay with 50 physical devices for different number of traffic sources

On the other hand, an increase of the number of physical devices, see Fig. 3b, has a strong impact on the SD value. In this case, 90 % of SD values for 2, 4 and 6 traffic sources is below 500 ms (leading to a ≈ ×100 increase as compared to the previous scenario) while the observed values with 8 flows would not mostly be acceptable.

In light of the results, some interesting conclusions can be extracted. The first one is that a hard constraint of the SD might not be always desirable, since the value of this parameter fluctuates along time, and the simulator might be able to recover from temporary maladjustment. We have also seen that, for the studied scenarios, while the number of traffic flows does not have a strong impact over the methodology accuracy, the number of physical devices within the scenario can definitively limit the validity of the emulation capabilities.

3.2 Evaluation of Video Applications

This section illustrates the evaluation of a real video streaming application, by using the emulation capability of ns-3. To this end, we have embedded instances of VLC player within LXC virtual devices. In order to have a finer control over the network, the coverage range of the WiFi nodes has been fixed to avoid collisions between packets belonging to different flows; in particular, we have used a disc propagation model with a maximum range of 15 m. It is worth highlighting that the 802.11 protocol implementation within the simulator does not prevent the different devices from contending for the channel, even if they are outside the coverage range of the transmitter (within the carrier sensing range).

First, we have carried out a number of simulations to study the effects that real applications might have over the emulation behavior. For that, we have

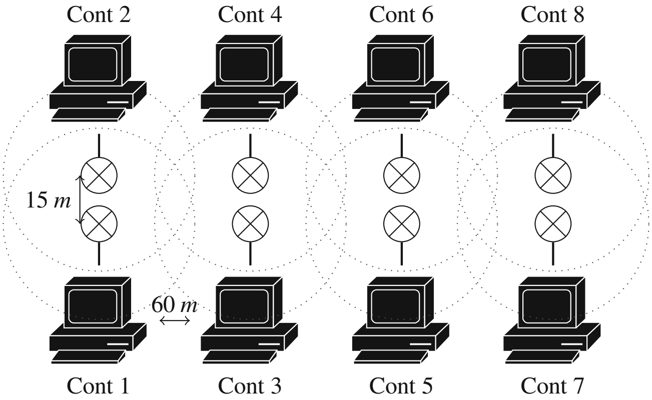


Fig. 4. General scenario with up to 4 video flows. The PCs represent Linux Containers that are attached to simulated WiFi nodes

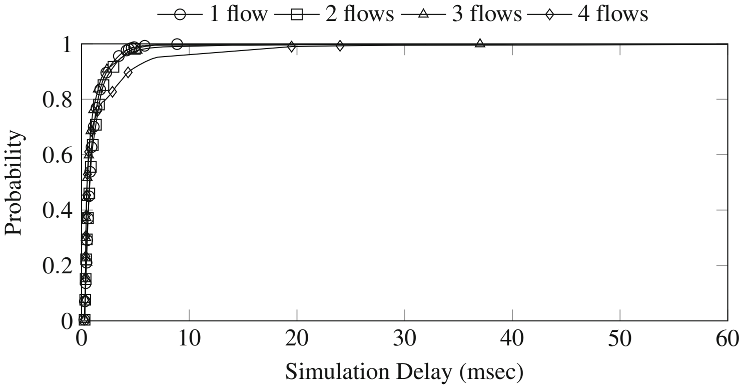


Fig. 5. Simulation delay with real video traffic

established a simple scenario in which the number of video flows is increased; Fig. 4 shows an illustrative example with 4 flows. As can be seen, each video flow implies a pair of LXC devices; furthermore, the various flows do not interfere with each other.

The performance metrics obtained from this scenario, but those corresponding to the SD, are always measured over the first pair of containers (1 and 2, see Fig. 4), allowing us to study the influence of the traffic load. Besides, 20 independent experiments of each configuration have been performed, ensuring the statistical validity of the results.

Figure 5 depicts the CDF of the SD parameter, considering different number of active flows. SD values have been again measured every 100 ms in each simulation. As can be seen, the simulator is able to carry out an appropriate emulation, with SD values below 30 ms up to when 4 traffic flows were active.

Comparing these results with those discussed previously, we can conclude that the number of devices deployed within the scenario has a stronger impact than the traffic load.

Once we have analyzed the performance of the proposed methodology when using real applications, i.e. with realistic traffic patterns, a number of illustrative QoS metrics are analyzed. First, Fig. 6 shows the application jitter, that has been measured as the gap between two consecutive packets at the receiver. The results reveal the shallow influence of the number of video flows over this metric, since all the values are very close to 0, as can be observed in Fig. 6a. Indeed, Fig. 6b, the zoomed version of Fig. 6a, shows that more than 90 % jitter samples are below 2.5 ms. The figure also yields the appearance of jitter peaks, that can be inferred from the small percentage of samples with values near 1 s.

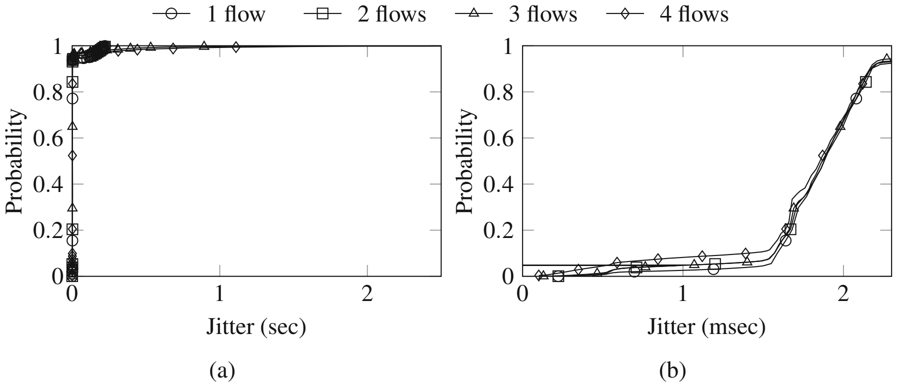


Fig. 6. CDF of the packet jitter upon different number of video flows

Afterwards, we have analyzed the end-to-end delay of the application packets, which is represented in Fig. 7. In this case, the results show that the delay increases as long as the number of traffic flows gets higher. As was previously said, all nodes generate events related to channel contention. This is clearly seen with the packet delay increase for the scenarios embracing 3 and 4 flows.

Finally, using the same scenario we have measured the data rate of the flows as they leave and enter the target containers. Figure 8 shows the CDF of the throughput experienced by the applications during one session; the throughput has been measured, alike the SD parameter, every 100 ms at both the transmission and reception sides of the first pair of containers. As can be seen, the presence of a larger number of video flows (more instances of video applications), has a clear impact on both the transmission and reception data rate. Since this particular parameter is measured at the simulator ‘entry-point’, we understand that the corresponding impact is due to the computer processing capacity.

So as to illustrate one of the advantages of the proposed methodology, we configure some of the wireless channels to induce errors. In this case, the scenario embraces a video server that, simultaneously, sends the same stream to two

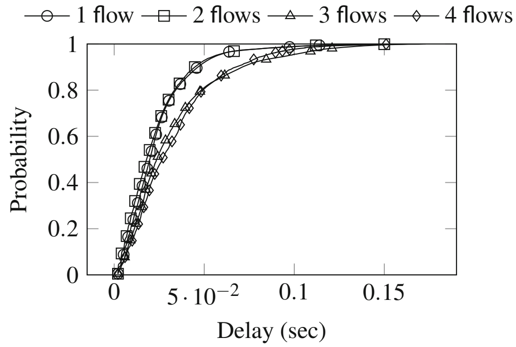
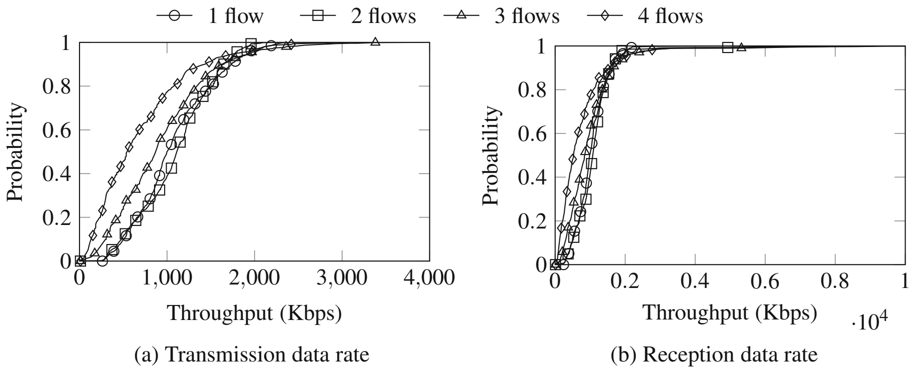


Fig. 7. CDF of the packet delay upon different number of video flows



(a) Transmission data rate

(b) Reception data rate

Fig. 8. CDF of the throughput of the first flow upon different number of video flows

clients. The scenario has been designed so that the channel conditions between the server and each client are different. Figure 9 illustrates the particular network topology; the video clients, **Client1** and **Client2**, require 4 and 2 hops to connect to the server, respectively. Besides, the channel conditions, that are established with the Frame Error Rate (FER), are worse in the route between the server and **Client1**. In this case, in order to analyze the behavior in a more realistic situation, we have carried out experiments lasting for several minutes.

Figure 10 depicts the throughput transmitted by the server and observed at each client. The results show that the throughput value at **Client2** is slightly higher than that measured at **Client1**; this is due to the more hostile channel conditions in this latter case (the corresponding path has more hops and the FER is also higher). Besides, the results also evidence that, while the throughput at the clients is rather constant (tightly around 1 Mbps), the application throughput at the server oscillates between 1 and 5 Mbps. It is worth highlighting that this kind of metric, as well as the packets sent by the application from the clients to the server, could not be correctly characterized using synthetic traffic models,

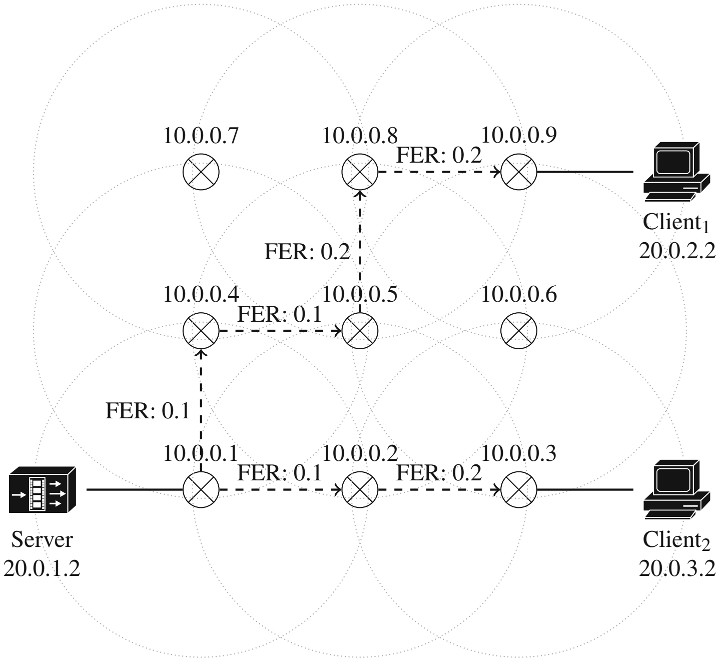


Fig. 9. Scenario with channel errors (FER)

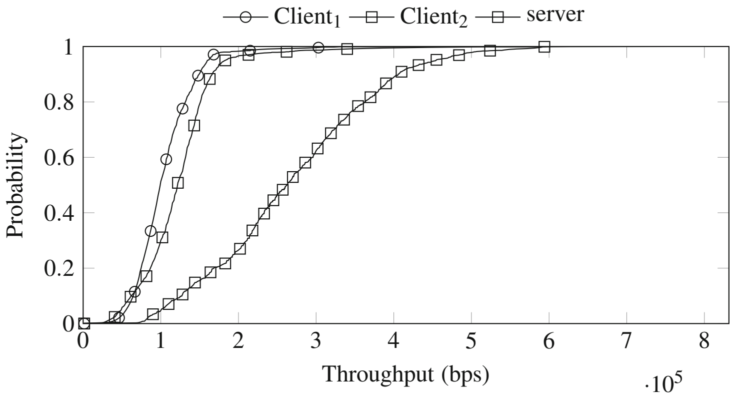


Fig. 10. CDF of the throughput experienced by the users and sent by the server

which are usually implemented in network simulators and traditionally used in various research works.

Figure 11 shows the percentage of packets that was received by each client, as well as the losses that were observed. As can be seen, in both cases there exists a significant packet loss, which is slightly higher for **Client1**, as a consequence of the worse conditions over the path it uses to connect to the server.

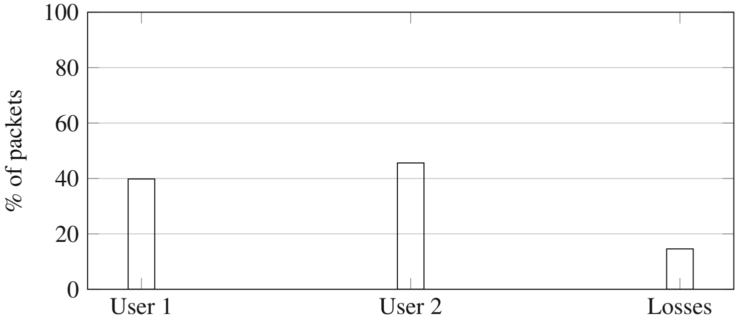
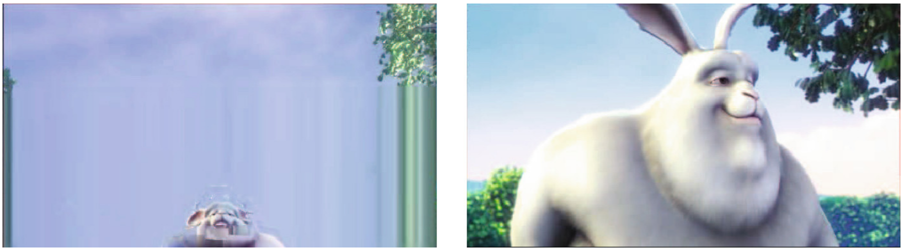


Fig. 11. Percentage of packets received by each client and total losses

Taking advantage of the functionality brought about by the methodology proposed in this work, it is possible to analyze, subjectively, the impact of the channel conditions, both errors and the number of required hops, on the user perception of the quality. In this sense, Fig. 12 depicts two snapshots, each of them belonging to the video flows received by the two clients, In general, moderated FER values (0.1 and 0.2) lead to a gradual degradation. The illustrative snapshots we have selected in Fig. 12 capture this general behavior, the quality of **Client1** is much worse. In general we can conclude that multi-hop wireless networks might have a rather negative impact over the performance of this type of applications.



(a) Client 1

(b) Client 2

Fig. 12. Example of video captures for both clients

4 Conclusion

In this work we have presented a methodology that promotes the use of emulation techniques to analyze video services over simulation platforms. It exploits the functionalities offered by the ns-3 tool, which allow its interaction with real systems. In particular, we have carried out an illustrative study on the behavior of real streaming video services when used over multi-hop wireless networks, using WiFi technology.

First, we have thoroughly studied the limitations exhibited by the simulator to appropriately tackle real-time emulation tasks, as well as other parameters that might impact the accuracy of the results that can be obtained. This first analysis yielded that, int he scenarios under study, the number of physical devices that are deployed in the corresponding scenario has a great impact over the capacity of the simulator to perform a real-time execution. On the other hand we have also observed that its influence is much stronger than the one due to a traffic load increase, since even those nodes not actively participating in the communications generate a relatively large number of events that the simulator needs to handle.

Besides, we have also seen some particular behaviors that might be attributed to the characteristics of the machine that was used to carry out all the experiments. In order to overcome them, a possible solution would be (when the number of active services is large) to transfer some of the applications to other (physical) hosts; this would help to alleviate the computational burden of the machine used to simulate the wireless network.

Finally, we have assessed the feasibility of the proposed methodology by studying the behavior of a real video service (using the VLC application), considering both ideal (error-free) wireless channels and the presence of errors induced by wireless propagation impairments. The obtained results show the behavior that might have been expected, validating the capability of the presented approach to carry out the evaluation of real applications over 'simulated' network deployments. We have indeed observed unexpected traffic patterns, which might not be reproduced with synthetic models (those traditionally used over simulation platforms), as another indication of the possibilities brought by this technique. It is important to mention that the approach ensures the possibility of systematically using a scenario with the very same conditions (reproducible in different experiments), providing it with a great flexibility, which was in fact one of the main pursued goals.

Regarding our future work, we are planning to exploit the proposed technique to broaden the analysis of different solutions over multi-hop wireless networks, including network coding, routing algorithms, etc., introducing real applications on top of them. On the other hand, as a more direct extension of the work presented herewith, we will also characterize the emulation capacity of the ns-3 platform, using other communication technologies, in particular LTE, due to its growing relevance.

Acknowledgment. This work has been supported by the Spanish Government by its funding through the project **COSAIF**, *"Connectivity as a Service: Access for the Internet of the Future"* (TEC2012-38754-C02-01).

References

1. CISCO: Visual Networking Index (VNI) Global Mobile Data Traffic Forecast Updateis part of the comprehensive Cisco VNI Forecast, an ongoing initiative to trackand forecast the impact of visual networking applications on global networks, February 2014
2. Network simulator 3. http://www.nsnam.org/
3. Howto make ns-3 interact with the real world. https://www.nsnam.org/wiki/HOWTO_make_ns-3_interact_with_the_real_world
4. Howto use linux containers to set up virtual networks. https://www.nsnam.org/wiki/HOWTO_Use_Linux_Containers_to_set_up_virtual_networks
5. Molloy, T., Yuan, Z., Muntean, G.-M.: Real time emulation of an lte network using ns-3. In: Irish Signals Systems Conference 2014 and 2014 China-Ireland International Conference on Information and Communications Technologies (ISSC 2014/CIICT 2014). 25th IET, June 2014, pp. 251–257 (2014)
6. Fouda, A., Ragab, A.N., Esswie, A., Marzban, M., Naser, A., Rehan, M., Ibrahim, A.S.: Real time video streaming over NS3 based emulated LTE networks. Int. J. Electr. Commun. Comput. Technol.(IJECCT) **4**(3) (2014)
7. Bustos-Jimenez, J., Alonso, R., Faundez, C., Meric, H.: Boxing experience: Measuring QoS and QoE of multimedia streaming using NS3, LXC and VLC. In: 2014 IEEE 39th Conference on Local Computer Networks Workshops (LCN Workshops), pp. 658–662, Sept 2014
8. Alvarez, A., Orea, R., Cabrero, S., Pañeda, X.G., García, R., Melendi, D.: Limitations of network emulation with single-machine and distributed ns-3. In: Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques, ser. SIMUTools 2010, ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), pp. 67:1–67:9 (2010). http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8630