How to Quantify *Trust* in Your Network Emulator?

Domenico Capriglione[1], Gianni Cerro[2](✉), Luigi Ferrigno[2], and Gianfranco Miele[2]

[1] DIIN, University of Salerno, Via Giovanni Paolo II 132, 84084 Fisciano, SA, Italy
dcapriglione@unisa.it

[2] DIEI, University of Cassino and Southern Lazio, Via G. Di Biasio 43, 03043 Cassino, FR, Italy
{g.cerro,ferrigno,g.miele}@unicas.it

Abstract. Network emulators are used in many contexts of communication networks for the design and the development of network management and routing strategies as well as for the tuning of multimedia services as Voice Over IP, video streaming, TV on-demand, to cite a few. These devices are generally used for modifying, in a controlled way, data traffic flows by changing, in real time, several critical parameters as delay, packet loss percentage, throughput, and so on. Due to very attractive features as high versatility and configurability and low cost, the solutions based on general purpose hardware platforms and free/open-source software are the most ones adopted in the practice for implementing network emulators. Nevertheless, in such architectures the complex interaction of software and hardware sections should affect the accuracy and repeatability of such systems in correctly emulating the desired network behaviors. Consequently, a suitable pre-characterization stage of such kind of network emulators should be performed before they are used. In this framework, the paper describes a methodological approach for designing suitable test-bed and measurement procedure able to reliably characterize the performance of such systems. The final aim of the research activity is to provide a suitable uncertainty model and a confidence level for the parameters provided by network emulators, which can drive the final users in more reliably analyzing the experimental results coming from their test campaigns and which involve the network emulators.

Keywords: Network emulators · Delay · Packet loss · Network measurements Metrological performance

1 Introduction

Computer and telecommunication networks are today involved in all main worldwide applications as telephony, financial transactions, banking, TV, on-demand entertainment services, Internet of Things, to cite a few [1–3]. In addition, the fourth industrial revolution, also known as Industry 4.0, pushes for bonding communication networks with the industrial ones with the general aims of improving the performance, the level of automation, the level of efficiency and quality of process and final products [4].

In these contexts, the design and management of communication networks are very important tasks because they straight affect the correctness, the quality and the reliability of the final services. Network active devices, algorithm routing strategies, communication protocols performance assessment are fundamental in order to assure that these services are correctly delivered to the final user by warranting the required level of expectation. These aspects are crucial also in modern and very promising Software Defined Networks (SDNs) [5].

Of course, due the complexity of the modern networks and the impracticability to make both development and testing on real scenarios, a widely used approach for these activities involves the use of suitable network emulators [6–8].

Unlike network simulation, where fixed models running on powerful computing devices try to evaluate complex systems behavior and to simulate real scenarios, the network emulation is able to introduce, in a controlled way, data traffic variations in real time [9, 10] for changing critical parameters as delay, packet loss percentage, throughput, and so on. By this way, performance devices/algorithms should be more reliably assessed before their deployment in actual networks. Obviously, the expected reliability improvement (with respect to simulators) can be achieved only if the emulator is able to accurately reproduce the desired network scenarios.

As for network emulators, on the basis of the architecture they can be divided in two main categories: Special Purpose Network Emulators (hereinafter SPNE) and General-Purpose Network Emulators (GPNE). The devices belonging to the former category are typically standalone devices, implemented on special purpose optimized and customized hardware and software platforms. Thanks to these characteristics, these solutions usually warrant very good performance, accuracy and repeatability even if they are characterized by high costs.

On the contrary, GPNE are based on computer programs to be run on general purpose hardware (i.e. Personal Computer) equipped with commercial interface cards and common operating systems. Among devices belonging to this category we can find both commercial and free/open-source software. Due to their cheapness and high degree of flexibility and versatility, the second ones are very widespread in practice [11] and, mainly thanks to the open-source feature, their employment is even more increasing also in the field of research. However, since they are based on general purpose platforms, their working strictly depends on the interaction of hardware and software sections of the hosting platform and their performance cannot be *a priori* guaranteed, as also proved in [12–15]. Therefore, prior to use such systems, a performance characterization and validation should be made for certifying the ability of these software to accurately reproduce the wanted network conditions. To this aim, a methodological approach for metrological performance evaluation of network emulators has not been adequately dealt in literature.

In this framework, focusing the attention on GPNE, starting from the past experiences in metrological characterization of computer networks devices and services [9, 15–18], a methodological approach is proposed to measure and analyze the performance of such systems. In a more detail, a suitable measurement setup is designed and characterized for analyzing the performance of GPNE. In order to show the application of the proposed methodology, two very popular network emulators, i.e. NetEM [11] and DummyNET [19] have been considered.

The paper is organized as follows: the proposed approach is reported in Sect. 2. The main features of the considered network emulators are described in Sect. 3 and an application example of the proposed approach is shown in Sect. 4. Finally, conclusions are provided in Sect. 5.

2 The Proposed Approach

To characterize the emulating ability of a GPNE, it is necessary to analyze the output of the network emulator when it is stressed with several inputs characterized by different traffic settings. To this aim a proper measurement set-up and measurement procedure are as illustrated in the next paragraphs. In particular, a three-step procedure is proposed (see Fig. 1).

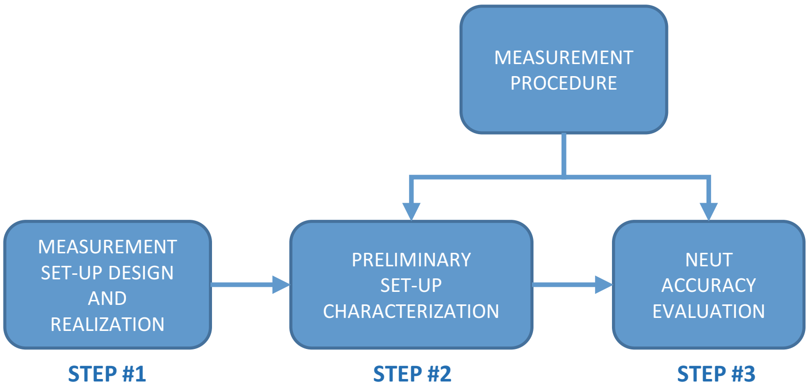
MEASUREMENT PROCEDURE
MEASUREMENT SET-UP DESIGN AND REALIZATION
PRELIMINARY SET-UP CHARACTERIZATION
NEUT ACCURACY EVALUATION
STEP #1
STEP #2
STEP #3

Fig. 1. The block diagram of the three-step procedure.

2.1 Measurement Set-Up Design and Realization

The proposed measurement set-up (i.e. step #1) is composed of three elements: a Reference Generator (RG), a Measurement Receiver (MR) and the network emulator under test (NEUT). As illustrated in Fig. 2, RG is connected to MR through the NEUT. RG provides a continuous service that creates a stream of packets towards the RG that acts as a sink. The characteristics of the traffic, like packet rate, packet loss, throughput and so on, can be selected by the user with aim of creating a stimulus for the NEUT. RM receives this stream of packets and measures the parameters of interest in order to quantify the accuracy of the NEUT in emulating a specific network characteristic.

2.2 Measurement Procedure

A sketch of the block diagram of the proposed measurement procedure is reported in Fig. 2. It represents the logical sequence of the operations executed during both preliminary set-up characterization (step #2) and NEUT accuracy evaluation (step #3).

Considering the importance of the clock synchronization of the host machine of the RG and MR, to avoid errors in the measurement of important parameters like packet delay, IPDV and so on, after a common initialization stage, NEUT is disabled, if it is operative.

After this operation, a clock synchronization procedure is executed between RG and MR inner clocks.

Once clock synchronization is over, NEUT is enabled. After this operation, MR is active in order to acquire and measure the incoming packets.

In the next step RG is enabled starting to generate a packet stream addressed to MR.

MR stores measurement data when RG has finished to generate the packet stream. The stored measurement data will be used to evaluate the parameters of interest related to the accuracy of the NEUT.

Finally, the measurement procedure is iteratively repeated *K* times with aim of improving the statistical significance of the obtained results.

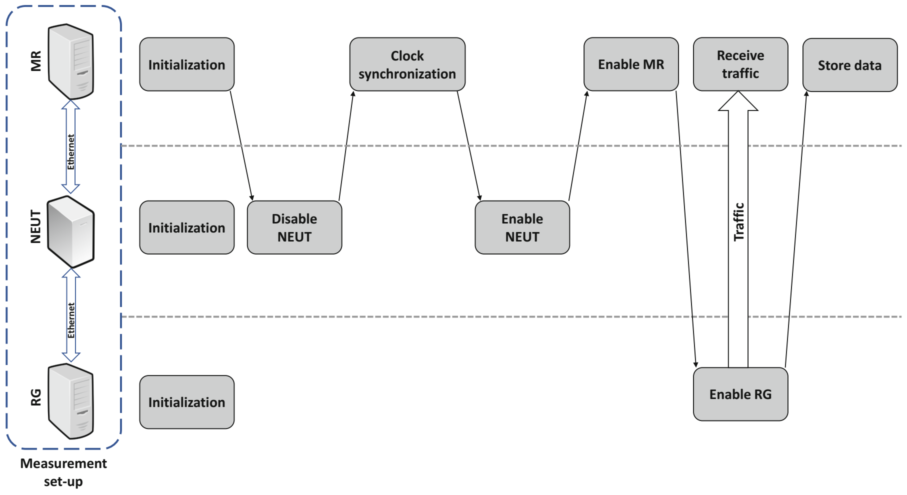

Fig. 2. Sketch of the proposed measurement set-up and procedure.

2.3 Preliminary Set-Up Characterization

To preliminary characterize the system (i.e. step #2), some tests with the emulators off have to be carried out. In particular, once selected the parameter to be tested by adopting the figures of merit described in the following subsection, the intrinsic non-idealities of the set-up itself and that cannot be imputable to the emulator capabilities are evaluated. This step allows evaluating the systematic effects introduced by the measurement set-up.

2.4 NEUT Accuracy Evaluation

To evaluate the accuracy of a NEUT in emulating a specific network condition (i.e. step #3), it is important to perform a statistical analysis of the parameters of interest, for example the experienced delay, the measured IPDV, packet loss and so on.

As a consequence, denoting with N the number of packets received during a test and $\xi_{i,j}$ the parameter of interest measured and related to the to the packet j in the i-th stream, the following quantities are evaluated:

$$\xi_i = \sum_{j=1}^{N} \frac{\xi_{i,j}}{N} \tag{1}$$

$$\mu_\xi = \sum_{i=1}^{K} \frac{\xi_i}{K} \tag{2}$$

$$\sigma_\xi = \sqrt{\frac{1}{K}\sum_{i=1}^{K}\left(\xi_i - \mu_\xi\right)^2} \tag{3}$$

$$m = \min \xi_i \tag{4}$$

$$M = \max \xi_i \tag{5}$$

where

- ξ_i is the average of the measures executed during the i-th test;
- μ_ξ is the mean value evaluated considering all the acquisitions;
- σ_ξ is the variability (standard deviation) of the mean value.

3 Considered NEUTs

In this paper we have considered two widely used GPNEs. They are free and open-source, namely NetEM and DummyNET.

3.1 NetEM

NetEM is the acronym of Network Emulator. It is a software tool, developed by Linux foundation, able to emulate the characteristics of a wide area network (WAN) such as delay, packet loss, etc. [10, 11]. It is in most Linux distributions with Kernels version 2.6 and higher and it is an extension of Linux traffic control. It allows to control and set the following traffic parameters: delay, packet loss, packet duplication, packet corruption and packet reordering.

Concerning delay, it can emulate fixed and random delay following different distributions (uniform, normal, Pareto and Pareto normal) and correlate successive delay values, trying to perform an imitation of congestion effects [10].

As packet loss regards, random loss of packets can be set by the user.

3.2 DummyNET

It is a tool developed at University of Pisa and originally designed for testing network protocols. It can emulate, delays, packet losses, multipath effects and bandwidth limitations. It runs on different operating systems like Windows, Linux and Mac OS.

It intercepts packets and passes them to object structures called *pipes*. A pipe represents a communication channel with a fixed-bandwidth and it implements a set of queues. Each queue emulates a packet queue in a network which delay or drop packets when congestion occurs [10].

4 An Application Example

In this section, with reference to the proposed three-step procedure described in Sect. 2, an application example is shown for analyzing and comparing the performance of the NEUTs described in Sect. 3. The obtained performance of adopted network emulators is provided as concerns the capability of emulating static delays and random packet losses. In detail, results are provided in terms of mean and standard deviation of the considered quantities over many trials for each test condition.

4.1 Measurement Set-Up Design and Realization

The hardware specifications of the proposed set-up are:

- RG and MR are PCs composed of: CPU Intel Pentium Dual Core E5400 @ 2.700 GHz, 4 GB RAM and a Network Interface Card (NIC) Atheros AR8121/AR8113/AR8114.
- NEUT is still a PC emulating a WAN having the same hardware characteristics of the former two PCs, but it is furthermore equipped with a second NIC Realtek RTL8169/8110.

The operating system (OS) installed on them is Scientific Linux 6.6. Each PC is connected to the WAN emulator through a 1.5 m length UTP category 5 cable. The network data rate has been set-up at 1000 Mb/s full duplex. To improve delay measurements accuracy, a NTP-based synchronization has been adopted.

4.2 Measurement Procedure

This subsection reports the measurement procedure parameters. Some details about their values (described in Sect. 2.2) are reported in Table 1.

Table 1. Numeric values of the experimental campaign parameters.

Feature	Type	Min	Max	Step
UDP traffic	Packet rate	100 pkt/s	5000 pkt/s	Logarithmic
UDP traffic	Duration	15 s	/	/
Delay	Static	1 ms	100 ms	Logarithmic
Packet loss	Random	1%	10%	Logarithmic
Test repetitions	No. of tests	50	/	/

In summary, traffic packets have been sent through the network with UDP protocol, the streaming transmission for each test lasted 15 s; different packet rates have been adopted to test the reliability of the NEUT under different stress conditions; during the tests, the NEUT has introduced several non-idealities, as described in Table 1, in terms of static delays and random packet losses. Delay and Packet Loss performance have been tested independently.

Furthermore, traffic generation, reception and packet feature extraction has been carried out by using software D-ITG [20].

4.3 Preliminary Set-Up Characterization

As described in Sect. 2.3 some tests with the emulators off have been carried out. The obtained delays and packet losses (i.e. the parameters to be tested) have been measured and reported. Such values constitute the intrinsic non-ideality of the set-up itself and they cannot be imputable to the emulator capabilities.

In Fig. 3, evaluation of experimental set-up intrinsic delay is reported. In particular, an error-bar-plot has been adopted to represent both the mean values (central points of each vertical bar, joint by an interpolating line) and standard deviations (the half of each vertical bar length), computed over the 50 trials for each test condition. It can be stated that the intrinsic delay introduced by the system itself is quantifiable in about 80 μs and the standard deviation is about 10 μs, if the worst case is considered. A decreasing trend is generally appreciable with respect to the packet rates. Best condition is achieved at packet rate equal to 2000 pkt/s, where the mean value is 71 μs and the standard deviation is equal to 5 μs. Anyway, by exploiting the concept of measurement compatibility, stating that two different measures are compatible if the intersection of their measurement intervals (evaluated as the numeric set obtained by adding and subtracting the extended measurement uncertainty to its mean value [21]) is not empty, it is possible to affirm that the intrinsic delay is not systematically changing at different packet rates and it can be considered constant and taken as the mean value

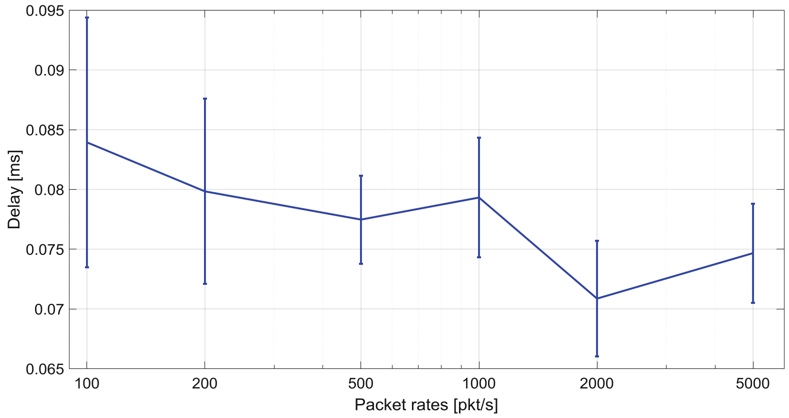

Fig. 3. Obtained delays under different packet rates with disabled emulator.

of delays obtained at different packet rates. This value should be generally taken into account (and eventually compensated) when testing the delays introduced by the NEUTs.

As for packet losses without emulator effect, we verified that the measurement system does not intrinsically introduce any problem related to packet loss, since no packets are lost in all tested conditions; therefore, following results, obtained with active emulator action, do not need to be normalized in terms of lost packets, unlike what happened for delay case.

4.4 NEUTs Accuracy Evaluation

In this section, results obtained by using the two considered emulators are reported. In particular, the emulators have both been set to introduce the same delays and packet losses to the network. Results are depicted in Figs. 4, 5, 6 and 7, and their presentation is carried out by reporting the mean values (μ in the graph legends) and the 2-coverage-factor curves (μ −2σ, μ +2σ), representing the trend of the mean values plus or minus the double of standard deviation values. Such boundary curves define the measurement interval where, under Gaussian hypothesis of the measurement distribution, the actual value resides with 95% probability.

4.4.1 Delay Related Results

In Figs. 4 and 5, the obtained delays with NetEM and DummyNET emulators are respectively reported. As stated in the introduction to this subsection, three curves for each imposed delay condition are shown in the figures. Due to the small values of standard deviations, in most cases such curves are superimposed, and a single curve is visible.

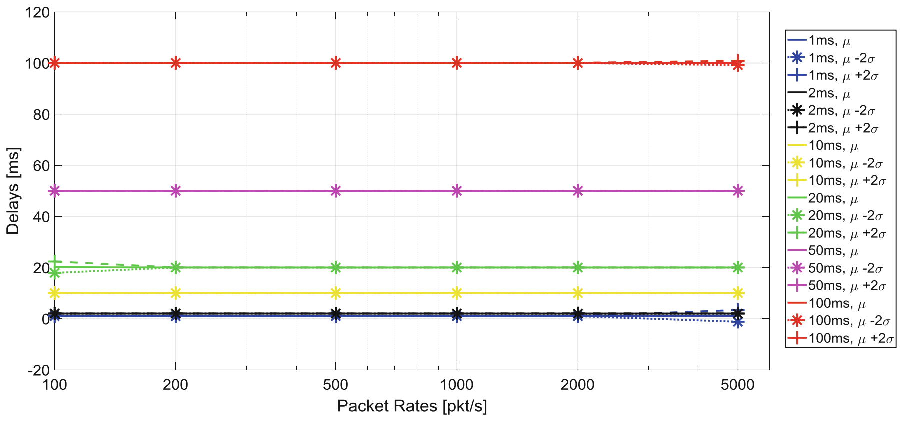

Fig. 4. Obtained delays under different packet rates with NetEM emulator.

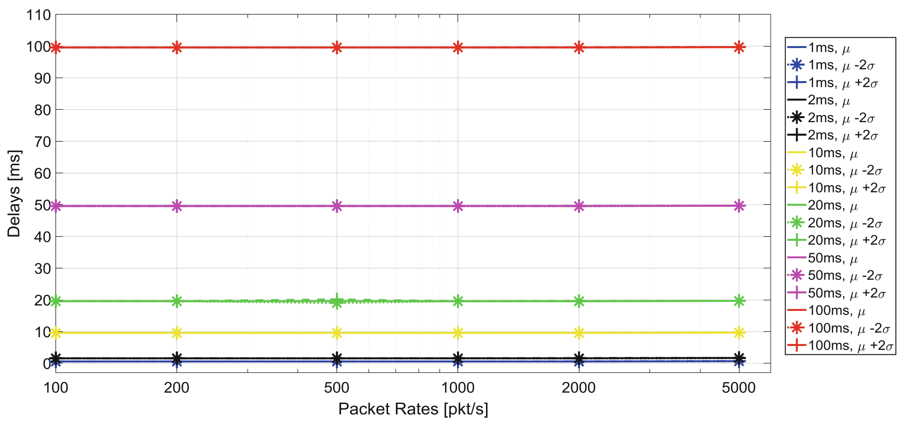

Fig. 5. Obtained delays under different packet rates with DummyNET emulator.

As expected, the mean values for each test condition are correctly reproduced, and they are pretty constant with respect to the packet rate. As regards NetEM (see Fig. 4), some particular behaviors can be observed for very low packet rate (100 pkt/s) at 20 ms imposed delay (green line) and at very high packet rate (5000 pkt/s) at 1 ms and 100 ms delay conditions (percentage standard deviations close to 5, 50 and 0.5%, respectively). In these cases, the repeatability of the measurement (intended as the capability to exhibit the same value under the same working conditions in repeated trials) is slightly worse than in other situations, where no distance among coverage and mean curves can be appreciated. Concerning DummyNET (Fig. 5), results are fully aligned with the expected values, and measurements are also highly repeatable (percentage standard deviations always lower than 0.5%), for all analyzed situations.

4.4.2 Packet Loss Related Results

Unlike delay performance, results concerning packet loss, reported in Figs. 6 and 7, show a particular behavior both with respect to the packet rate variation and repeatability. In detail, a common trend of both emulators is the poor capability to keep the packet loss constant and aligned with the imposed values, especially for high value of packet rates and required packet loss. The observed decreasing trend leads to have a measured packet loss equal to 8%, whilst the imposed one is 10% both for NetEM and DummyNET. Such phenomenon is not appreciable for lower values of packet losses, where the constant trend is kept for all packet rates condition. The second particular phenomenon related to packet loss is the very poor repeatability of the measurement results, revealed by the distance of the mean value and coverage curves, especially visible in low packet rates conditions. The standard deviations become lower in high packet rate cases.

This behavior has an important consequence: when packet rate is set to 100 pkt/s and imposed packet loss is 1 or 2%, the obtained measurement intervals are partially superimposed, thus providing compatibility among such measurements. This result

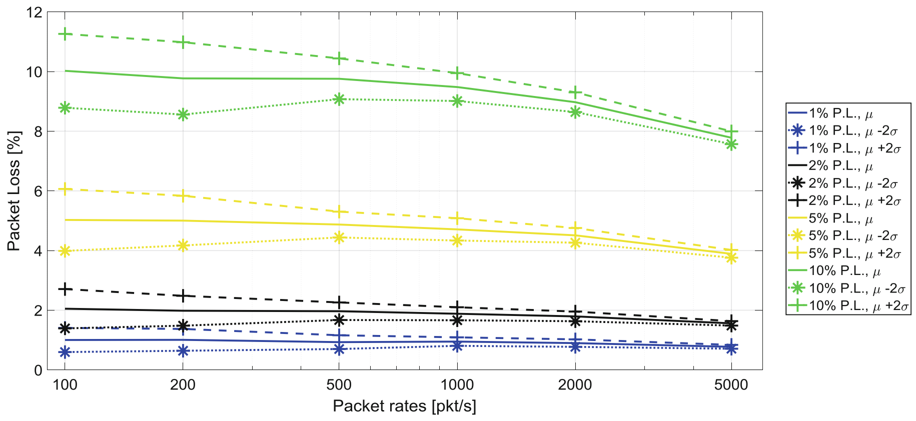

Fig. 6. Obtained packet losses under different packet rates with NetEM emulator.

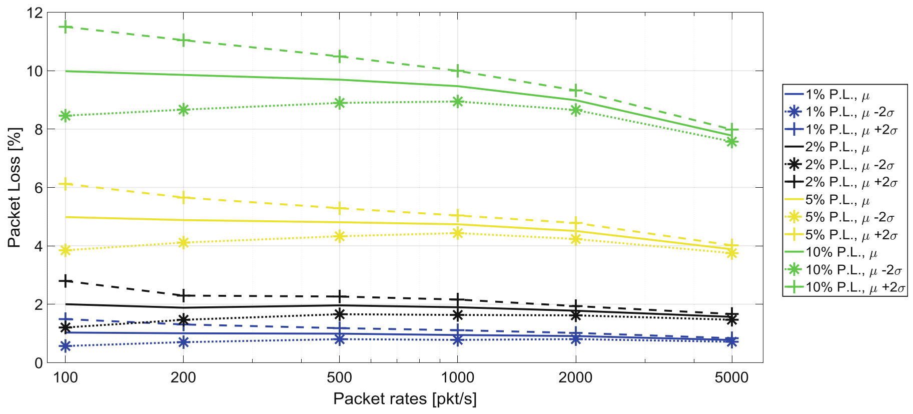

Fig. 7. Obtained packet losses under different packet rates with DummyNET emulator.

implies that if one takes the measurement in those points cannot predict if the imposed packet loss was 1 or 2%, because obtained behaviors are similar.

5 Conclusions

In this paper, a methodological approach for characterizing the performance of network emulators based on general purpose platforms and free/open source software is described. In particular, the measurement setup, the procedures for its preliminary characterization and for carrying out the performance measurements are explained. To show the application of the proposal, it has been applied to real case studies for analyzing the performance of two popular network emulators, very widespread in the academic research context. In particular, their ability in providing reliable packet loss percentages and delays has been evaluated for different values of such quantities (i.e. different imposed settings) and working conditions in terms of packet rate. The proposed approach reveals as useful also for comparing in a systematic way the metrological performance of such kind of network emulators.

Further developments will concern with the extension of the proposed approach to further parameters typically considered in network emulation, as Throughput and Internet Packet Delay Variation (together with their probability density functions) and further network emulators with the aim of better identifying factors that influence the accuracy of the emulated traffic. In addition, the possibility of providing a simple accuracy model of the parameter emulated as a function of quantities of influence will be investigated as well.

References

1. Coyle, C.L., Vaughn, H.: Social networking: communication revolution or evolution? Bell Labs Tech. J. **13**, 13–17 (2008). https://doi.org/10.1002/bltj.20298
2. Angrisani, L., Narduzzi, C.: Testing communication and computer networks: an overview. IEEE Instrum. Meas. Mag. **11**(5), 12–24 (2008). https://doi.org/10.1109/MIM.2008.4630738
3. Ayele, E.D., Meratnia, N., Havinga, P.J.M.: Towards a new opportunistic IoT network architecture for wildlife monitoring system. In: 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Paris, France, pp. 1–5 (2018). https://doi.org/10.1109/NTMS.2018.8328721
4. Lucas-Estañ, M.C., Raptis, T.P., Sepulcre, M., Passarella, A., Regueiro, C., Lazaro, O.: A software defined hierarchical communication and data management architecture for Industry 4.0. In: 14th Annual Conference on Wireless On-demand Network Systems and Services (WONS), Isola 2000, France, pp. 37–44 (2018). https://doi.org/10.23919/WONS.2018.8311660
5. Abdallah, S., Elhajj, I.H., Chehab A., Kayssi, A.: A network management framework for SDN. In: 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Paris, France, pp. 1–4 (2018). https://doi.org/10.1109/NTMS.2018.8328672
6. Koshimura, R., Ito, Y.: Development of Web-QoE evaluation system for wireless LAN with combination of simulator and network emulator. In: IEEE 2nd Global Conference on Consumer Electronics (GCCE), Tokyo, pp. 431–432 (2013). https://doi.org/10.1109/GCCE.2013.6664879
7. Holt, C., Kong, A., Leger, A.S., Bennett, D.: Communications network emulation for smart grid test-bed. In: IEEE Power and Energy Society General Meeting (PESGM), Boston, MA, pp. 1–5 (2016). https://doi.org/10.1109/PESGM.2016.7741999
8. Lal, C., Laxmi V., Gaur, M.S.: Video streaming over MANETs: testing and analysis using real-time emulation. In: 19th Asia-Pacific Conference on Communications (APCC), Denpasar, pp. 190–195 (2013). https://doi.org/10.1109/APCC.2013.6765940
9. Angrisani, L., Capriglione, D., Cerro, G., Ferrigno, L., Miele, G.: Experimental analysis of software network emulators in packet delay emulation. In: IEEE International Workshop on Measurement and Networking (M&N), Naples, pp. 1–6 (2017). https://doi.org/10.1109/IWMN.2017.8078382
10. Beuran, R.: Introduction to Network Emulation. Pan Stanford Publishing Pte. Ltd .(2013). ISBN: 9789814364096

11. Hemminger, S.: Network Emulation with NetEm (2005). Inlinux.conf.au
12. Hoßfeld, T., Fiedler, M.: The unexpected QoE killer: when the network emulator misshapes traffic and QoE. In: Seventh International Workshop on Quality of Multimedia Experience (QoMEX), Pylos-Nestoras, pp. 1–6 (2015). https://doi.org/10.1109/QoMEX.2015.7148093
13. Beshay, J.D., Francini, A., Prakash, R.: On the fidelity of single-machine network emulation in linux. In: IEEE 23rd International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, Atlanta, GA, pp. 19–22 (2015)
14. Betta, G., Capriglione, D., Ferrigno, L., Laracca, M.: A Measurement driven approach to design an efficient test methodology for PLT network QoS performance parameters assessment. Meas. Sci. Technol. (2009). https://doi.org/10.1088/0957-0233/20/10/105101
15. Angrisani, L., Botta, A., Miele, G., Pescapé, A., Vadursi, M.: Experiment-driven modeling of open-source internet traffic generators. IEEE Trans. Instrum. Meas. **63**(11), 2529–2538 (2014). https://doi.org/10.1109/TIM.2014.2348633
16. Angrisani, L., Capriglione, D., Ferrigno, L., Miele, G.: Internet Protocol Packet Delay Variation measurements in communication networks: how to evaluate measurement uncertainty? Measurement **46**(7), 2099–2109 (2013). https://doi.org/10.1016/j.measurement.2013.03.007
17. Angrisani, L., Capriglione, D., Ferrigno, L., Miele, G.: A methodological approach for estimating protocol analyzer instrumental measurement uncertainty in packet jitter evaluation. IEEE Trans. Instrum. Meas. **61**(5), 1405–1416 (2012). https://doi.org/10.1109/TIM.2012.2186478
18. Angrisani, L., Capriglione, D., Ferrigno, L., Miele, G.: An internet protocol packet delay variation estimator for reliable quality assessment of video-streaming services. IEEE Trans. Instrum. Meas. **62**(5), 914–923 (2013). https://doi.org/10.1109/TIM.2013.2245051
19. Carbone, M., Rizzo, L.: DummyNet revisited. SIGCOMM Comput. Commun. **40**(2), 12–20 (2010). https://doi.org/10.1145/1764873.1764876
20. Botta, A., Dainotti, A., Pescapé, A.: A tool for the generation of realistic network workload for emerging networking scenarios. Comput. Netw. **56**(15), 3531–3547 (2012). https://doi.org/10.1016/j.comnet.2012.02.019
21. JCGM: Evaluation of measurement data—guide to the expression of uncertainty in measurement. JCGM 100 (2008)