information

Article

Software-Driven Definition of Virtual Testbeds to Validate Emergent Network Technologies †

David Muelas *, **Javier Ramos and Jorge E. López de Vergara**

High Performance Computing and Networking Research Group, Departamento de Tecnología Electrónica y de las Comunicaciones, Escuela Politécnica Superior, Universidad Autónoma de Madrid, Francisco Tomás y Valiente, 11, 28049 Madrid, Spain; javier.ramos@uam.es (J.R.); jorge.lopez_vergara@uam.es (J.E.L.d.V.)

* Correspondence: dav.muelas@uam.es; Tel.: +34-914-972-291

† This paper is an extended version of our paper published in XIII Jornadas de Ingeniería Telemática (JITEL 2017), Valencia, Spain, 27–29 September 2017, "Definición de Testbeds Virtualizados Utilizando Perfiles de Actividad de Red".

Received: 9 January 2018; Accepted: 21 February 2018; Published: 24 February 2018

Abstract: The lack of privileged access to emergent and operational deployments is one of the key matters during validation and testing of novel telecommunication systems and technologies. This matter jeopardizes the repeatability of experiments, which results in burdens for innovation and research in these areas. In this light, we present a method and architecture to make the software-driven definition of virtual testbeds easier. As distinguishing features, our proposal can mimic operational deployments by using high-dimensional activity patterns. These activity patterns shape the effect of a control module that triggers agents for the generation of network traffic. This solution exploits the capabilities of network emulation and virtualization systems, which nowadays can be easily deployed in commodity servers. With this, we accomplish a reproducible definition of realistic experimental conditions and the introduction of real agent implementations in a cost-effective fashion. We evaluate our solution in a case study that is comprised of the validation of a network-monitoring tool for Voice over IP (VoIP) deployments. Our experimental results support the viability of the method and illustrate how this formulation can improve the experimentation in emergent technologies.

Keywords: network virtualization; software-defined networks; reproducible experimentation; virtual testbeds; performance evaluation; network behavior replication; voice over IP

1. Introduction

The advent of novel telecommunication technologies and applications is posing an accelerated evolution of the behavior of current operational networks [1]. As a result, classic network simulation tools, such as OMNeT++ or ns-3, cannot successfully suit emergent scenarios on many occasions. Moreover, these simulation tools are not intended to generate realistic network load or traffic traces, with complex applications in execution that generate data. Consequently, their applicability is severely restricted during the evaluation of algorithms, architectures or applications that require this type of data. This is particularly notorious for research in disruptive infrastructures or novel management paradigms; e.g., Internet of Things (IoT), fog computing or Software-Defined Networking (SDN) [2]. These challenges motivate the exploration of novel approaches that overcome the limitations of such widespread experimentation tools.

In this regard, physical testbeds arise as one of the most popular alternatives to network simulation tools. There is a wide variety of experimental platforms that offer several hardware resources and grant different access privileges to users; see, for example, the reviews in [3–5]. Nonetheless, while physical testbeds provide an invaluable resource, they also present several issues during experimental processes. For instance, huge deployments present scalability issues and high expenditures, and differences in access policies and privileges may constrain the reproducibility. Furthermore, the usual lack of control of the underlying infrastructure can induce additional uncertainty in the results.

In line with these facts, many recent research efforts have tried to exploit the opportunities that Network Function Virtualization (NFV) brings. Unsurprisingly, the flexibility and reduced cost of this approach are key aspects that explain such interest. As a matter of fact, several tools have transformed how network elements are operated and defined following this trend. On the one hand, many vendors (for instance, see [6,7]) are including virtual nodes in commercial network equipment to ease the introduction of complex network functions inside switching or routing elements. In this light, network functions are abstracted as software pieces that run on top of hypervisors and that can access network traffic traversing the equipment. On the other hand, several solutions extend the virtualized elements in general-purpose Operating Systems (OS), to provide abstractions of network elements. For example, Mininet [8,9] is a platform that allows the instantiation of virtual switches with OpenFlow support by using network namespaces of GNU/Linux systems. Its easy operation, open source orientation and modest requirements have converted Mininet into one of the reference tools for reproducible research in SDN.

Following the philosophy that inspired this latter approach, we present an architecture and method to generate complex synthetic activity profiles that mimic the macroscopic behavior of operational networks. As distinguishing features, our solution: (i) is able to reproduce non-stationary dynamics, which can be easily defined in terms of typical behavior baselines; and (ii) relies on open source software elements that can be deployed on commodity hardware. Additionally, we present empirical evidence of the viability of such an approach.

Our proposal aims to define risk-free testing environments with low expenditures, which can substantially improve experimental processes in the networking scope. Furthermore, this approach can bring even more advantages if applied to the study of emergent scenarios, as researchers usually face severe difficulties when accessing them; see, for example, the situations related to the study of wireless sensor networks reviewed in [3]; or the experiences related to experimentation with SDN testbeds compiled in [10–12].

To accomplish such objectives, our solution makes use of software-driven configurations for agents in charge of the generation of network traffic in virtual testbeds. These virtual testbeds can be easily deployed in commodity servers using any virtualization technique, following the same principles that guide NFV. In this manner, this design decomposes the definition of a virtual testbed into a topology instantiation, a description of the macroscopic behavior of the network and a software definition of the traffic flows that agents generate. Therefore, our solution can help open access to software-driven testbeds where the experimental conditions are formally described.

The evaluation of our proposal is two-fold. On the one hand, we analyze the activity patterns that it generates in a use case for the validation of a passive network monitoring tool for Voice over IP (VoIP) traffic analysis. On the other hand, we qualitatively compare the outcomes of this process with other alternatives, as previous tools cannot be directly compared to ours. In such a manner, we present robust evidence of the advantages that our proposal brings.

The rest of this paper is organized as follows. In Section 2, we present a review of several related works. To do so, we consider several matters that motivate our study and methodological approach; a review of the technologies that make our approach feasible; and summarize some previous solutions for synthetic network load generation. Section 3 includes the description of our proposal, both in a general version and with some adaptations to mimic VoIP deployments. Section 4 presents the results of our experimental evaluation and discuss the implications of our proposal. Finally, Section 5 highlights the main conclusions and findings of our work.

2. Related Work

This section is devoted to previous results that motivate and support our proposal. To do so, we have structured our review into three parts. The first part considers results that expose the importance that testing in realistic scenarios deserves. The second one describes several technologies that have been used to deploy virtual network elements in commodity servers, thus enabling the development of frameworks and solutions similar to ours. Finally, the third encompasses works that have explored the synthetic generation of network traffic. They can be considered the grounding of our solution and, at the same time, a complementary toolbox that can be enhanced with our method.

2.1. Background

The evaluation of novel methods, algorithms and tools in the networking scope requires the availability of testbeds that fairly represent the actual operation of infrastructures. Otherwise, the results of their deployment may be far from the ones achieved in the controlled scenarios.

This is particularly critical when the experimental setups themselves hide potential error conditions. An illustration of such a matter is reported in [13], where the authors showed that high performance capture engines suffer from traffic timestamping errors that cannot be detected when the evaluations are restricted to their typical worst-case scenarios.

However, the validation of new solutions in operational networks is a challenging process, given the importance that these infrastructures have for companies. In this light, the introduction of dynamics that mimic their behavior in experimental setups can mitigate the shortcomings of isolated testing, while keeping the risk of failure in operational environments low.

Additionally, validation in such environments usually entails hard restrictions related to data disclosure and protection. Taking into account the review in [14], we summarize different ethical issues for research in ICTs in Table 1. While these issues are linked to one or more aspects for the protection of individuals, they can dramatically limit the repeatability of results.

Remarkably, these aspects can be avoided when using synthetic datasets that do not included data directly gathered from users. In fact, we claim that a macroscopic replication of the behavior may suffice in many cases where real data are used; e.g., experimental processes in which the methodology focuses on the performance of protocols or technologies related to services with hard anonymity constraints such as the case of VoIP [15]. Hence, the restrictions related to ethical issues can be relaxed if only a description of such behavior is needed and disclosed.

Table 1. Ethical issues for research in ICTs.

Ethical Issue	Research Design Aspects
Privacy	Boundaries to preserve anonymity of underlying individuals and definition of safeguards and actor(s) in charge of this protection.
Accuracy	Guarantees related to the gathered information and definition of who is responsible for these guarantees.
Property	Owner(s) of both artifacts and collected information.
Access	Definition of authorizations and conditions for actor(s) making use of gathered information and designation of responsibilities related to this use.

To face these matters, our solution provides fine-grained data and exploits virtualization elements to improve scalability, reduce cost and minimize the impact of ethical issues on network experimentation.

2.2. Enabling Technologies

The exploitation of virtualization platforms for network testing has been explored since the popularization of virtual machines. For example, works such as [16,17] studied the suitability of the use of such elements for the evaluation of multimedia applications. Nonetheless, while these results share some common principles with our proposal, the technological characteristics of the proposed solutions are far from the current trends in network virtualization and emulation.

For these reasons, the applicability of these previous proposals may be currently limited, which motivates the development of novel techniques and tools.

Particularly, the exploitation of lightweight virtualization technologies and NFV [18,19] has arisen as a promising field of innovation in the study of computer networks [20]. We summarize the capabilities of several approaches in Table 2. They provide different levels of isolation among the threads corresponding to each virtualized network element, with a consequent cost in terms of resource consumption. Table 2 also includes some examples of platforms making use of each virtualization approach, which have been selected for illustrative purposes given their popularity among researchers.

Table 2. Summary of enabling technologies and platforms for the definition of virtual testbeds.

Virtualization	Key Features	Platforms
Virtual Machines (VMs)	• Each host is a VM and executes its own OS. • OS may differ among hosts.	• VNX (Virtual Networks over linuX) • VM managers
Containers	• Each host is represented with one or more containers. • Host processes are isolated, but OS kernel is not virtualized.	• Docker • VNX • IMUNES (Integrated Multiprotocol Network Emulator/Simulator)
Namespaces	• Each host is represented with a namespace/cgroup (control group to limit, account and isolate usage of resources). • Each host instantiates its processes, but processes are not isolated.	• Mininet • Maxinet

Reported use cases (such as those presented in [21,22] for Mininet, [23] for IMUNES (Integrated Multiprotocol Network Emulator/Simulator) or [24,25] for VNX (Virtual Networks over linuX) show the suitability of this type of tool for the study of operational networks, as long as certain accuracy loss in the results is acceptable [1,26]. Moreover, studies such as [11,12] expose how these tools can enhance the experimentation in emergent network environments.

All these results motivate the analysis of the possibilities offered by network emulation platforms, in order to access controlled environments that are able to represent trustworthy future network deployments.

2.3. *Synthetic Network Load Generation*

Despite the opportunities that they offer, the automatic generation of network load in scenarios defined on network virtualization and emulation platforms is still a challenging issue, as it must be representative of the scenarios under test. We have selected the works in [27–30], as they are representative examples of existent synthetic traffic-generation systems. Such approaches make use of parameters extracted from traffic traces to provide realistic traffic flows.

In [27], the authors defined a method for generating connections with similar statistical characteristics to those present on real network traffic. In [28], a comprehensive survey on network traffic load generation is presented. Such a survey motivates the architectural design decisions that are relevant for this type of system, some of which share ideas with the solution presented in this work. Recently, the work in [29] analyzed the characteristics of data flows extracted from specific sources, and [30] described the implementation of a high-performance IP generation solution based on FPGAs.

The evaluation of such solutions proved that the characteristics of synthetic traffic can be statistically indistinguishable from real traffic. However, our proposal focuses on replicating the dynamics of macroscopic network activity [31,32] using agents to generate fine-grained datasets. Hence, our approach starts with the analysis of active connections, in order to define complex aggregated behaviors. This feature eases the evaluation of new protocols and tools, by generating synthetic traffic based on the observed activity of real applications. Thus, our proposal offers a novel functionality that enriches the aggregated behavior of the generated network load and that, to the best of our knowledge, is not included in any previous tool.

3. Formulation of the Method for the Generation of Network Activity

As stated before, our solution uses software agents that emulate the activity of users to generate realistic network activity automatically. This feature allows us to split the definition of virtual testbeds into: (i) the topological specification of interconnected elements; (ii) a control plane for the activity of the agents in charge of traffic generation; and (iii) the specific characteristics of such traffic.

Figure 1 illustrates our architecture proposal, which follows this approach. The functional elements that appear in the architecture include a control node that configures and activates the traffic-generation agents in the remaining nodes. Finally, an additional capture and analysis node is included to analyze the global aggregated behavior. This design ensures that both configuration and activity generation are related only to a single application. In another case, the proposed architecture can be replicated as many times as needed for each different application that has to be included in the testbed, and a further aggregation of the resulting traffic can be performed. In the following, we describe how we derive control policies to activate specific traffic-generation agents that produce packets aiming at the usefulness in a wide set of situations.

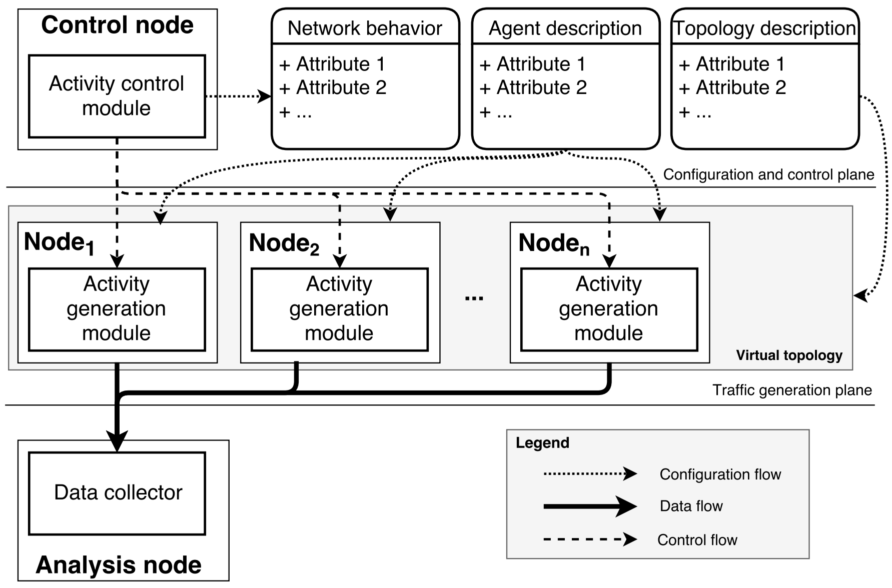


Figure 1. Diagram of the architecture of our proposal.

3.1. General Traffic-Generation Model

First of all, to generate the traffic activity, our method requires that the following basic hypothesis are met:

- A baseline for the temporal evolution of the number of active connections exists.
- A characteristic process for the creation of new connections per time unit exists and can be adjusted using its expected value.
- The distribution of the connections' duration does not change over time.

The first hypothesis entails that there exists a high-dimensional baseline, which can be defined with a function $F(t), t \in \mathbb{T} \subset \mathbb{R}$. $\mathbb{T}$ is the compact set that represents the time domain of the baseline; e.g., a daily period. This hypothesis emphasizes that a somehow stable network activity pattern must exist if network dynamics have any invariants. In fact, sustained changes in such invariants may point to changes in the usage of the network [33], which contradicts the idea of representing the operational state of an infrastructure.

Nonetheless, in our setup, this baseline can be inferred from the behavior of a real network [31,32] (i.e., to replicate it) or can be fixed to adapt testing scenarios to specific situations under test; we will use the latter approach in our case study. This dual orientation makes the repetition of trials in scenarios that mimic an operational situation of interest easier; but also the introduction of custom patterns to complement benchmarks.

Regarding the connections' duration, our model considers that it is the time period in which the connection is active on the network. In other words, queuing effects are not considered, so it equals the duration of the network flow from the standpoint of a passive element that receives all the traffic. Additionally, from the third hypothesis, we can infer that the expected value for the connection duration (W) must be constant.

Using $F(t)$ and W, we want to define an approximation to the expected value for the new connection process in order to modulate the activity of the testbed and adjust it to the expected dynamic. Following the proof of Little's law [34], we use a decomposition of the compact $\mathbb{T}$ in a succession of compacts $\{\mathbb{T}_i\}$ such that its union is $\mathbb{T}$ and they are pairwise disjoint. As W is constant, it is possible to define the expected number of new connections based on Equation (1):

$$\lambda(t) = \frac{\sum_{t \in \mathbb{T}_i} F(t)}{W}, \ \forall t \in \mathbb{T}_i \tag{1}$$

The new connection process $A(t), t \in \mathbb{T}$ is adjusted in such a way that Equation (2) holds:

$$\mathbb{E}[A(t)] = \lambda(t), \forall t \in \mathbb{T} \tag{2}$$

Finally, the control node configures a random number of new connections between the hosts present in the virtual topology. Such a random number is generated using the process adjusted in each time step. The new connections are activated, established and maintained in an autonomous way by the load-generation nodes.

3.2. Specific Traffic-Generation Model for VoIP Devices

To adapt the general model previously presented to the VoIP traffic-generation scenario, let us consider the classic telephony model. In such a model, the number of new connections $A(t)$ follows a Poisson distribution for which Equation (3) holds:

$$A(t) \sim \mathrm{Poi}(\lambda(t)), t \in \mathbb{T} \tag{3}$$

and the duration of connections follows an exponential distribution for which Equation (4) holds:

$$S_{\mathcal{C}} \sim \mathrm{Exp}(1/W(t)), t \in \mathbb{T} \tag{4}$$

for all the connections $\mathcal{C}$ created at time t.

Regarding the connections, two data flows must be generated for each direction of a VoIP call: one corresponding to the signaling and another corresponding to the multimedia data. The combination of signaling and multimedia transport protocols has been selected in such a way that represents common environments both in domestic and enterprise networks. Particularly, the signaling protocols used are the Session Initiation Protocol (SIP) [35] and the Skinny Client Control Protocol (SCCP) [36]. The multimedia transport protocol selected is the Real-time Transport Protocol (RTP) [37] using the payload type corresponding to the G.711 codec [38].

4. Experimental Evaluation

In what follows, we provide a complete description of the hardware, proof of concept implementation and activity profiles considered during the execution of our experiments. After that, we report the results obtained during the deployment of these elements in our case study. The software used for the load generation is available by request for any person interested in its use or modification.

4.1. Methodology

Let us now focus on the methodological aspects of our tests. All the experiments have been executed in a Commodity Off-The-Shelf (COTS) server equipped with two Intel Xeon E5-2620 processors with six cores per processor with an operation frequency of 2.10 GHz and 32 GB of RAM memory. To avoid uncontrolled effects produced by hardware virtualization, hyper-threading characteristics have been disabled. The server operating system is Ubuntu 14.04.01 with a 4.4.0-45 Linux kernel. For illustrative purposes, Mininet [8,9] Version 2.2.2. (available at https://github.com/mininet/mininet) with the default configuration has been used for all the tests.

The diagram of the virtual testbed is presented in Figure 2. This diagram displays all the links for data transmission (configured without any further limitation, such as packet loss or delay) and the logical structure of control and configuration connections. The deployments that we used during the experimental evaluation of the method included 150 hosts, as our intention was to replicate a Local Area Network (LAN). Only one host includes the agent that activates traffic generation agents, and one virtual switch (Open vSwitch) interconnects the remaining elements. Finally, one host receives all the traffic and runs an instance of VoIPCallMon [39,40], to detect and analyze the generated traffic; in fact, this setup was used during the validation of this tool before deploying it in a network with a similar activity pattern.

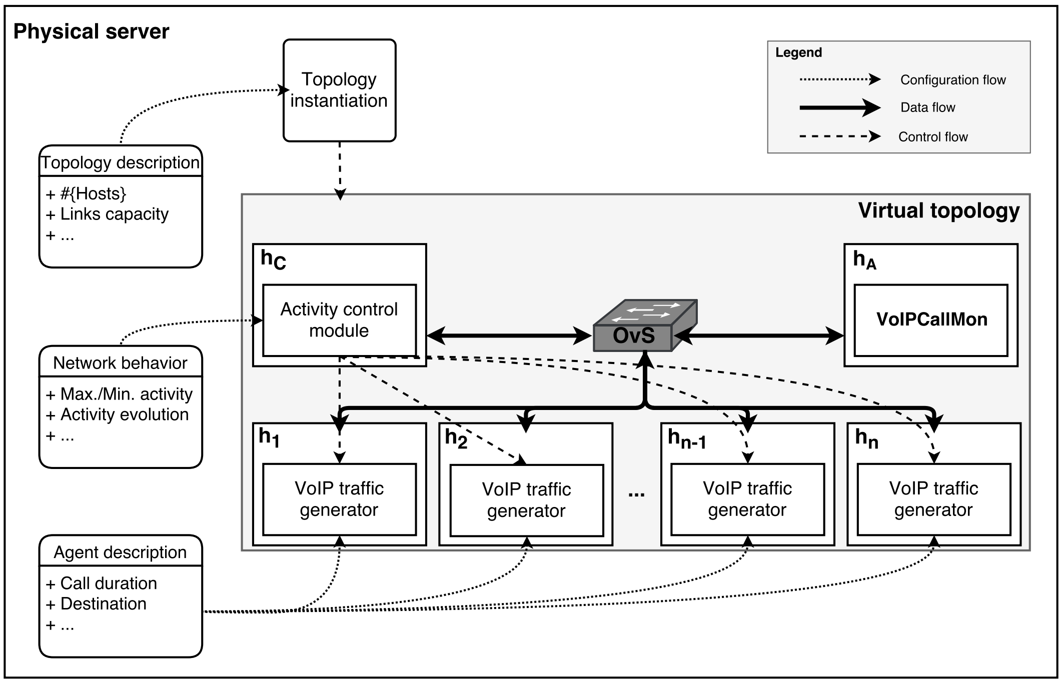


Figure 2. Virtual topology defined for the experimental setup in the case study.

4.2. Resource Consumption

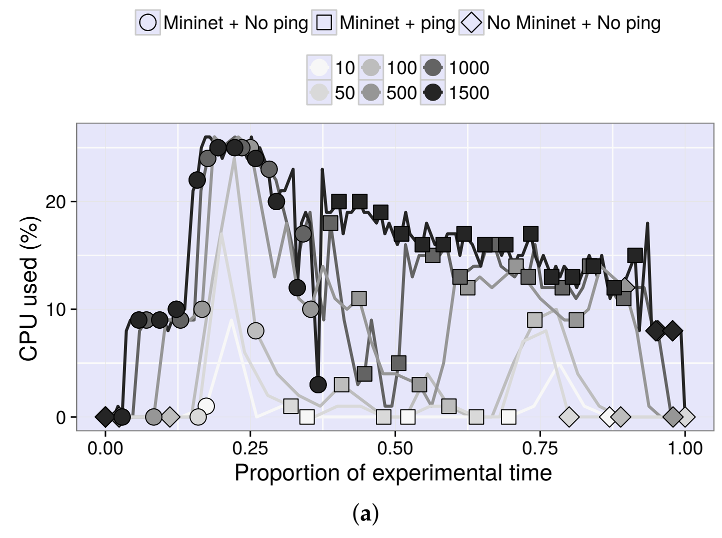
Before running the networks emulated during the case study, we evaluate the viability of such an approach by stressing a virtual network deployment. We focus on two key resource consumption metrics, namely percentage of CPU used and memory occupancy (Figures 3 and 4). These resources define hard boundaries in terms of the number of processes that can be instantiated and executed on the hardware system.

We monitor them in environments with low and high network activity (using `ping` or `iperf` in hosts, respectively) and varying the number of hosts: from 10–1500. With this experimental setup, we cover all the situations that may arise in the case study in the following sections.

For the percentage of CPU used, we note that during the first stages of virtual topology instantiation (denoted as "Mininet + No ping") CPU consumption is below 25%. Interestingly, this consumption is mainly produced by the initialization and interconnection of virtual switches in the topologies.

In the scenarios with low network activity (i.e., executing `ping` in the hosts), CPU load substantially decreases as packet sending and reception (hence, the instructions and system calls in the hosts) are low; specifically, `ping` is sending one packet per second.

However, in the case of high network activity (i.e., executing `iperf` in the hosts), the percentage of CPU used reaches 100% in many cases, as a result of network transmissions with a traffic rate near 10 Gb/s. Unsurprisingly, in these cases, the effect of adding further hosts is much more significant than in the low network activity scenarios. This is a direct consequence of the interaction among the number of elements and the activity in each one of them.



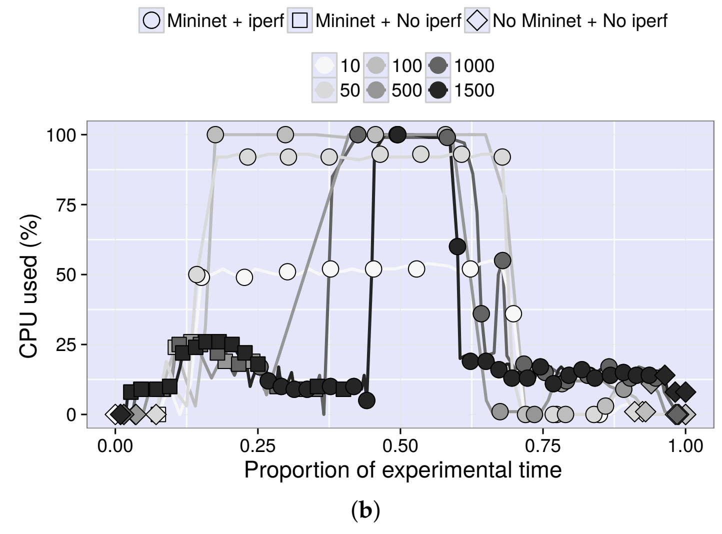
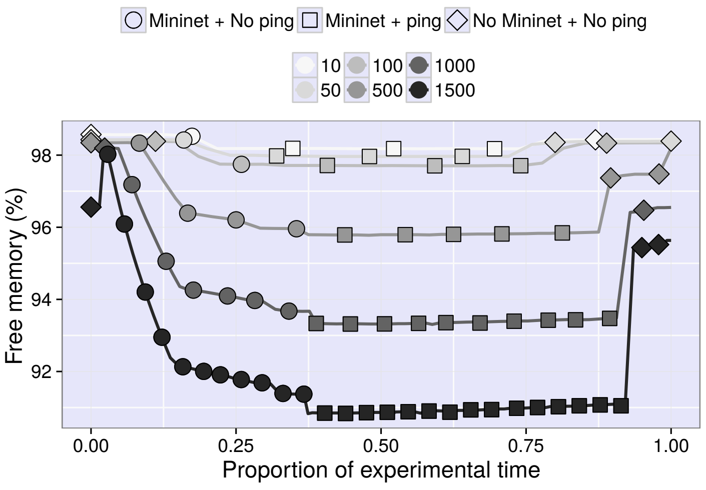


Figure 3. Percentage of used CPU in terms of number of hosts, 10 switches. (**a**) Low network activity; (**b**) high network activity.

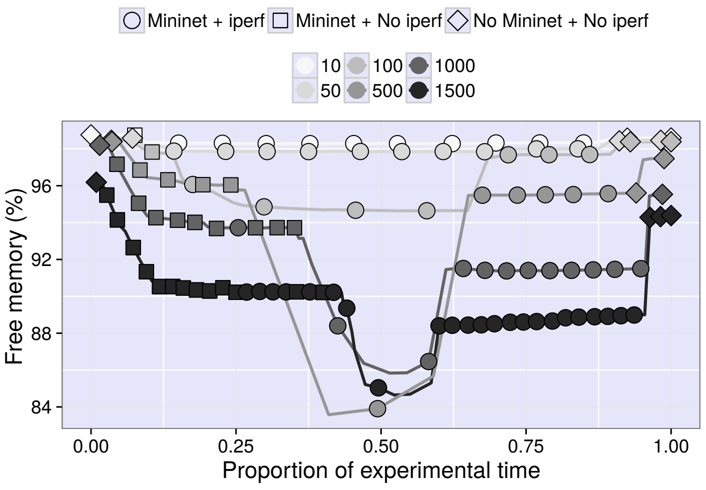
Memory consumption presents similar patterns: during the instantiation of virtual topologies, the percentage of used memory grows with the number of active hosts. However, the consumption is still moderate even for the biggest topology size: 1500 hosts. In particular, it is below 16% in all the experiments and does not reach 10% of the total memory when no processes are executed in the virtual hosts.

Regarding the effect of network activity, we note that `ping` exerts an almost undetectable change in memory consumption, while `iperf` causes a higher additional consumption of memory: around 12% in the most intensive scenario. Interestingly, memory occupancy in the intervals of highest activity does not present a clear relation to the number of active hosts.

These experiments show that the (i) CPU use is highly linked to network activity and processes in virtual hosts and (ii) memory consumption depends mainly on the number of instantiated hosts. In this light, the instantiation of a virtual topology with up to 1500 hosts and 10 switches does not require more than 20% of the total memory in our server and that moderate to medium network loads will not saturate CPUs. Therefore, these results prove that our proposal can run on top of Mininet in a general-purpose server.



(**a**)



(**b**)

Figure 4. Percentage of free memory in terms of number of hosts, 10 switches. (**a**) Low network activity; (**b**) high network activity.

4.3. Measured Network Activity

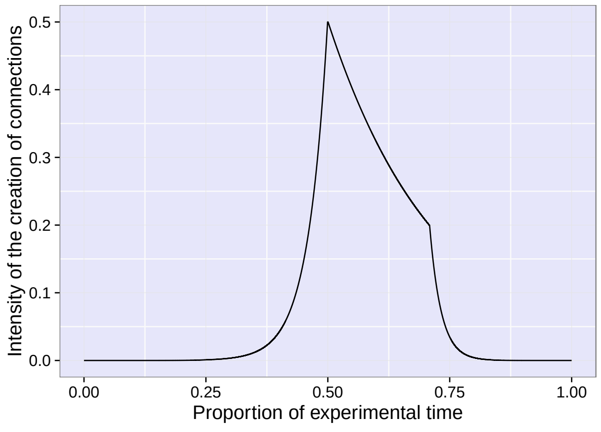
In our case, the objective is to study the functionality and stability of VoIPCallMon when monitoring a local VoIP network in realistic situations. Needless to say, gaining access to a real VoIP telephone network for the evaluation of a monitoring system is challenging, as a result of the consequent personal data disclosure, which raises privacy and access issues as stated in Section 2.1. For this reason, we set the requirement of 50 concurrent calls in the maximum activity period, and the generated traffic was tuned to show the typical activity pattern of an enterprise network; that is, with a busy period located near 12 noon, a fast increment during the opening and some spurious connections outside of the working hours.

Hence, our goal is to apply the previously-defined method to generate 24-h activity profiles that mimic such deployments. To generate this pattern, we preset two different profiles of the daily concurrence of connections based on previous VoIP monitoring experiences. This evaluation highlights the flexibility of our method, in terms of its generality in mimicking different dynamical profiles, and depicts two typical activity trends for VoIP networks: offices with a narrower activity period, e.g., administrative services, and with a wider one, e.g., call centers, respectively.

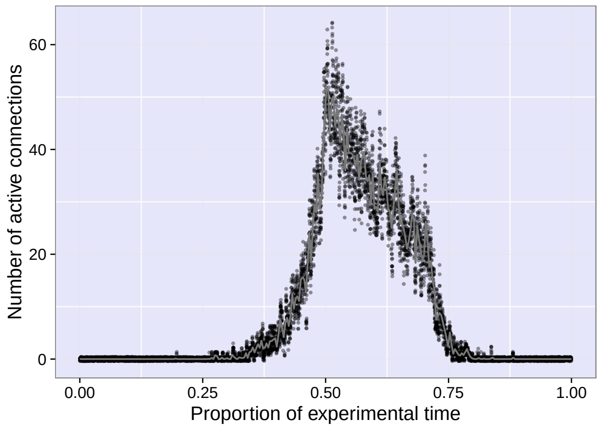
The first profile is parametrized by four temporal points:

- Activity starting time: 6 a.m.
- Activity end time: 8 p.m.
- Busy hour (H1): 12 p.m.
- Transition from mid-afternoon to close period (H2): 5 p.m.

Additionally, we fix the concurrence at H1 and H2 to follow the expected number of connections. With these points, we define a piecewise function that follows the defined profile, and we transform it to obtain the new connection (new calls) pattern as shown in Figure 5a.



(**a**) Intensity of the creation of connections.



(**b**) Active connections.

Figure 5. Analysis of generated load. (**a**) represents the intensity models for new connections; (**b**) represents the resulting connections: average activity in 5 min is displayed with a gray line.

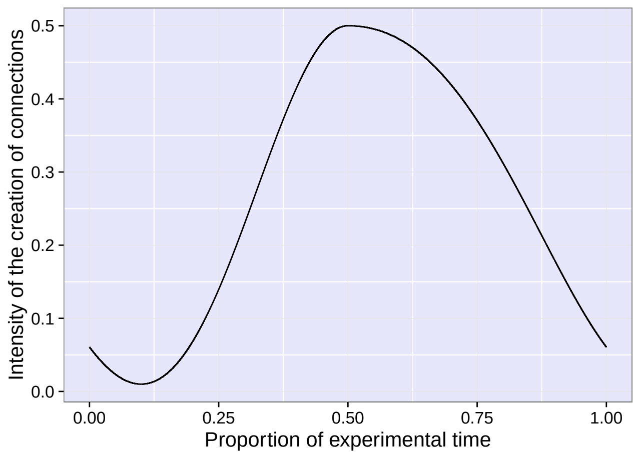
The second profile depends on two temporal moments:

- High activity (H1): 12 p.m.
- Low activity (H2): 3 a.m.

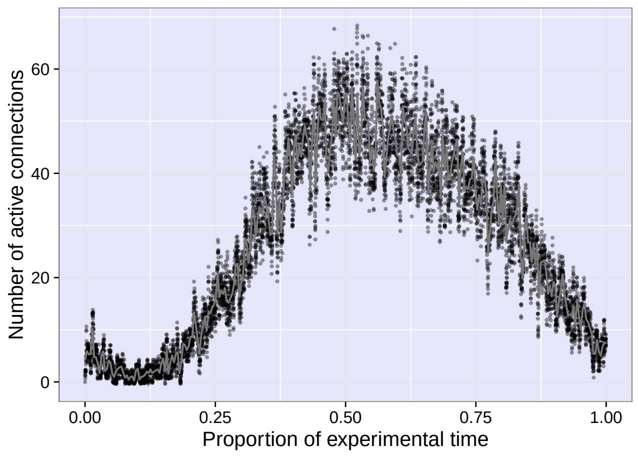
These two values are related to high and low activity periods, respectively. As with the first profile, in this case, we also fix the concurrence values in such moments and obtain the pattern depicted in Figure 6a.

Using the method described in Section 3, we generate activity that replicates the profile for an activity day, and we analyze the aggregate traffic received using VoIPCallMon. The traffic generation is performed using the Scapy Python library due to its simplicity for building network packets. Scapy eases the generation of SCCP packets, as it is a proprietary VoIP signaling protocol implemented in Cisco terminals. On the other hand, the generated calls used the G.711 codec for audio transmission, generating a packet every 20 ms of samples. Based on real traffic analysis in enterprise networks, we have selected 100 s as the average call duration, which is expectable in office environments.

Figures 5b and 6b represent the trajectories (i.e., realizations of the underlying stochastic process) detected by this tool, corresponding to the first and second profile respectively. The activity fairly fits the requirements in both cases, as the average activity (presented as a gray line inside the cloud of points) manifests.



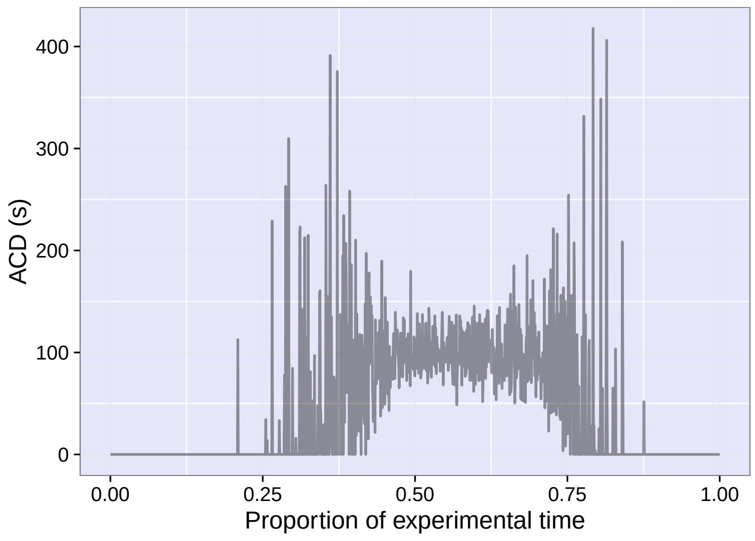
(**a**) Intensity of the creation of connections.



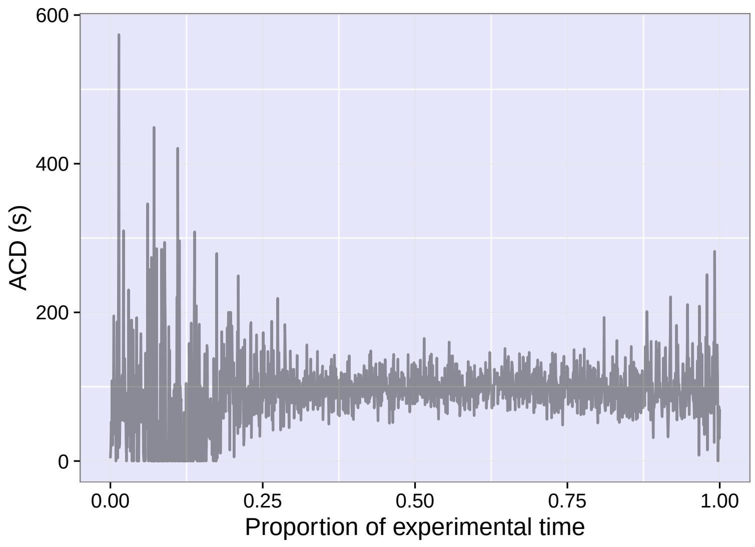
(**b**) Active connections.

Figure 6. Analysis of generated load. (**a**) represents the intensity models for new connections; (**b**) represents the resulting connections: average activity in 5 min is displayed with a gray line.

Additionally, we monitor the Average Call Duration (ACD) in the system as it may be used as a QoS parameter. The results along a trajectory are presented in Figure 7a,b, for each of the activity profiles. As expected, the ACD converges to the specified value in the model construction (100 s) in the periods of higher load, with higher variation in low activity ones.



(**a**) ACD behavior along a daily trajectory.



(**b**) ACD behavior along a daily trajectory.

Figure 7. Analysis of the Average Call Duration (ACD). (**a**,**b**) represent the ACD for the first and second profiles, respectively.

Furthermore, by simulating different trajectories over the domain $\mathbb{T}$, we can observe that activity in equivalent instants in different realizations of the dynamic shows a stable behavior adapted to the previously-defined parameter; Figure 8 illustrates such idea over 20 s of the busy period of 40 trajectories, showing boxplots for each day and the mean function. The variation of the mean function is explained by the standard deviation of the new connection-generation process; equal to the mean, due to the properties of the Poisson distribution.

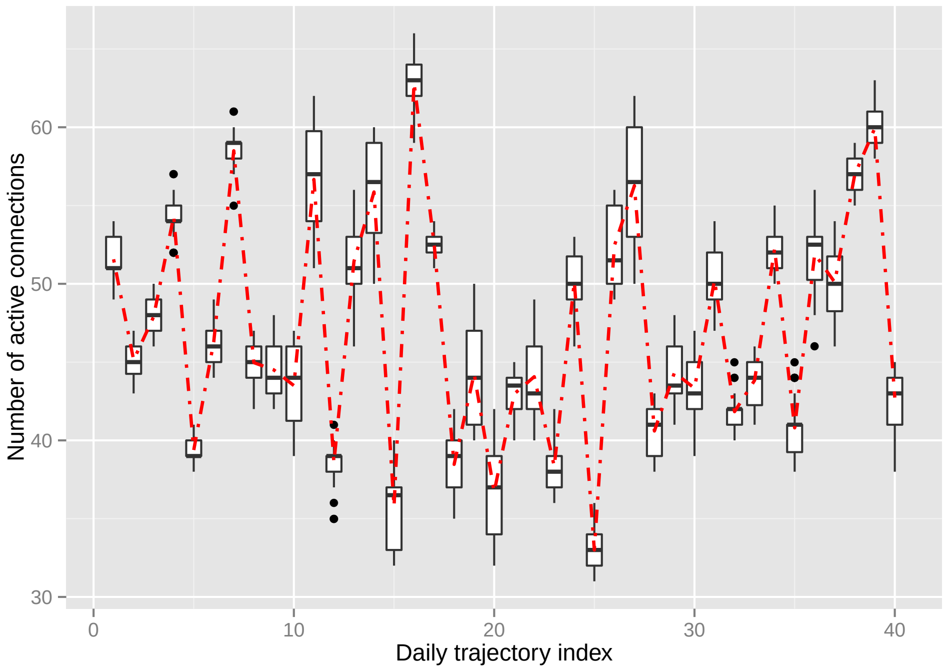


Figure 8. Evolution of the connections in 30 s of the busy period, 40 trajectories. Each boxplot shows the values for the same day, and the red dashed line shows the mean function.

4.4. Qualitative Comparison with Other Methods

We have shown that our method is able to generate activity for users from a macroscopic definition of network behavior. In other words, it can be applied to replicate network behaviors once their activity patterns are extracted from aggregated real measurements. As a result, our proposal can substitute other approaches during experimental processes.

Remarkably, the specific design of our solution provides some distinguishing qualitative features when comparing with other state of the art testing approaches. We have synthesized some key aspects in Table 3, which puts together the general capabilities, risks, ethical issues and cost of each approach. In this table, we denote with the symbol "-" aspects that depend on specific situations or conditions.

Table 3. Qualitative comparison of testing approaches.

	Capabilities			**Risks**		**Expenditures**
Approach	**Realistic Behavior**	**Fully Customizable**	**Network Traffic Available**	**Risk of Critical Failure**	**Light Ethical Constraints**	**Cost**
Operational network	✓	✗	✓	High	✗	Low
Real traffic traces	✓	✗	✓	Low	✗	Low
Hardware testbed	✗	-	-	None	✓	High
Network simulation	-	✓	✗	None	✓	Low
Our proposal	✓	✓	✓	None	✓	Low

As major outcomes of our case study, the application of our solution surpassed the absence of the access to both VoIP networks and traffic traces because of ethical and privacy issues. Additionally, it made feasible the testing of a monitoring tool that needs to capture network traffic, thus improving the results obtained via simulations. Finally, our solution did not incur the high economic cost of an equivalent hardware testbed, as the execution of the virtual testbed only required a COTS server instead of deploying hundreds of phones and VoIP gateways.

5. Conclusions

Along this work, we have presented a proposal for the generation of network load that replicates the behavior of an operational deployment. Such a proposal can improve the development of experimental studies that require network traffic and activity, by enabling the definition of virtual testbeds making use of macroscopic characteristics. Therefore, our proposal relaxes the ethical concerns related to information disclosure and can help to improve the repeatability of a broad variety of experimental setups.

The technical viability of our proposal is rooted in the feasibility of virtualizing network elements and functions. The software-driven configuration of agents connects the activity in the network elements with high-dimensional profiles that represent specific scenarios. We have shown how such an approach performs in a commodity server when implemented using Mininet. The figures of resource consumption support the viability of this solution.

To validate the improvements that our proposal provides, we have reported the results of a case study to evaluate a real monitoring tool for VoIP analysis in a virtual testbed. We have illustrated the traffic generated when two different activity profiles are configured, showing how the measured network activity fits them.

Additionally, we have qualitatively compared our proposal with other widespread approaches. The results of this comparison highlight the gain that our solution brings to the experimentation in the networking scope.

The results in this paper can foster the development of tools to evaluate other emergent scenarios in risk-free and easily reproducible environments while keeping expenditures low. In this light, our future work lines aim to extend the presented case studies to other applications and scopes. We are currently exploring some promising areas in the study of multimedia transmissions in scenarios with drones and IoT nodes; by deploying virtual topologies with activity trends that reflect typical multimedia transmissions from nodes and monitoring elements such as we did in the presented case study. Additionally, we are also adding support for the integration of SDN elements and traffic engineering tools such as the `tc` command of GNU/Linux systems, to make the development of complex scenarios with a realistic traffic load easier, which is generated following our proposal.

Acknowledgments: This work has been partially funded by the Spanish Ministry of Economy and Competitiveness and the European Regional Development Fund under the projects TRÁFICA (MINECO/FEDER TEC2015-69417-C2-1-R) and RACING DRONES (MINECO/FEDER RTC-2016-4744-7).

Author Contributions: David Muelas and Javier Ramos designed the model, system architecture and experimental setup, analyzed the results, and wrote the paper. Jorge E. López de Vergara offered background knowledge, part of the literature review and supervision of the results and paper writing.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pediaditakis, D.; Rotsos, C.; Moore, A.W. Faithful Reproduction of Network Experiments. In Proceedings of the Tenth ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS '14), Los Angeles, CA, USA, 20–21 October 2014; pp. 41–52.
2. Xia, W.; Wen, Y.; Foh, C.H.; Niyato, D.; Xie, H. A Survey on Software-Defined Networking. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 27–51.
3. Horneber, J.; Hergenröder, A. A Survey on Testbeds and Experimentation Environments for Wireless Sensor Networks. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1820–1838.
4. Goel, U.; Wittie, M.P.; Claffy, K.C.; Le, A. Survey of End-to-End Mobile Network Measurement Testbeds, Tools, and Services. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 105–123.
5. Tsai, P.W.; Piccialli, F.; Tsai, C.W.; Luo, M.Y.; Yang, C.S. Control frameworks in network emulation testbeds: A survey. *J. Comput. Sci.* **2017**, *22*, 148–161.
6. NFX250 Network Services Platform. Available online: https://www.juniper.net/us/en/products-services/sdn/nfx250/ (accessed on 23 February 2018).

7. Arista 7500E Series. Available online: https://www.arista.com/en/products/7500-series (accessed on 23 February 2018).
8. Lantz, B.; Heller, B.; McKeown, N. A network in a laptop: Rapid prototyping for Software-Defined Networks. In Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, Monterey, CA, USA, 20–21 October 2010; ACM: New York, NY, USA, 2010; pp. 1–6.
9. Yan, J.; Jin, D. VT-Mininet: Virtual-time-enabled Mininet for Scalable and Accurate Software-Define Network Emulation. In Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research (SOSR '15), Santa Clara, CA, USA, 17–18 June 2015; pp. 1–7.
10. Handigol, N.; Heller, B.; Jeyakumar, V.; Lantz, B.; McKeown, N. Reproducible Network Experiments Using Container-based Emulation. In Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies (CoNEXT '12), Nice, France, 10–13 December 2012; pp. 253–264.
11. Lantz, B.; O'Connor, B. A Mininet-based Virtual Testbed for Distributed SDN Development. *SIGCOMM Comput. Commun. Rev.* **2015**, *45*, 365–366.
12. Baldesi, L.; Maccari, L. NePA TesT: Network protocol and application testing toolchain for community networks. In Proceedings of the 12th Annual Conference on Wireless On-demand Network Systems and Services (WONS), Cortina d'Ampezzo, Italy, 20–22 January 2016; pp. 1–8.
13. Moreno, V.; del Río, P.M.S.; Ramos, J.; Garnica, J.J.; García-Dorado, J.L. Batch to the Future: Analyzing Timestamp Accuracy of High-Performance Packet I/O Engines. *IEEE Commun. Lett.* **2012**, *16*, 1888–1891.
14. Myers, M.D.; Venable, J.R. A set of ethical principles for design science research in information systems. *Inf. Manag.* **2014**, *51*, 801–809.
15. Meeran, M.T.; Annus, P.; Alam, M.M.; Moullec, Y.L. Evaluation of VoIP QoS Performance in Wireless Mesh Networks. *Information* **2017**, *8*, 88.
16. Bachmeir, C.; Tabery, P.; Uzumcu, S.; Steinbach, E. A scalable virtual programmable real-time testbed for rapid multimedia service creation and evaluation. In Proceedings of the 2003 International Conference on Multimedia and Expo (ICME '03), Baltimore, MD, USA, 6–9 July 2003; Volume 3, pp. 257–260.
17. Fuertes, W.; López de Vergara, J.E. An emulation of VoD services using virtual network environments. *Electron. Commun. EASST* **2009**, *17*, doi:10.14279/tuj.eceasst.17.224.220.
18. Han, B.; Gopalakrishnan, V.; Ji, L.; Lee, S. Network function virtualization: Challenges and opportunities for innovations. *IEEE Commun. Mag.* **2015**, *53*, 90–97.
19. Mijumbi, R.; Serrat, J.; Gorricho, J.L.; Bouten, N.; Turck, F.D.; Boutaba, R. Network Function Virtualization: State-of-the-Art and Research Challenges. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 236–262.
20. Peach, S.; Irwin, B.; van Heerden, R. An overview of linux container based network emulation. In Proceedings of the European Conference on Information Warfare and Security (ECCWS), Munich, Germany, 7–8 July 2016; pp. 253–259.
21. Raza, M.; Chowdhury, S.; Robertson, W. SDN based emulation of an academic networking testbed. In Proceedings of the 2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Vancouver, BC, Canada, 15–18 May 2016; pp. 1–6.
22. Rong, R.; Liu, J. Distributed mininet with symbiosis. In Proceedings of the 2017 IEEE International Conference on Communications (ICC), Paris, France, 21–25 May 2017; pp. 1–6.
23. Fontes, R.D.R.; Mahfoudi, M.; Dabbous, W.; Turletti, T.; Rothenberg, C. How Far Can We Go? Towards Realistic Software-Defined Wireless Networking Experiments. *Comput. J.* **2017**, *60*, 1458–1471.
24. Lentisco, C.M.; Aguayo, M.; Bellido, L.; Pastor, E.; De-Antonio-Monte, D.; Bolívar, A.G. A virtualized platform for analyzing LTE broadcast services. In Proceedings of the 2015 European Conference on Networks and Communications (EuCNC), Paris, France, 29 June–2 July 2015; pp. 512–516.
25. Moyano, R.F.; Cambronero, D.F.; Triana, L.B. A user-centric SDN management architecture for NFV-based residential networks. *Comput. Stand. Interfaces* **2017**, *54*, 279–292.
26. Jimenez, J.M.; Martínez, J.O.R.; Rego, A.; Dilendra, A.; Lloret, J. Study of multimedia delivery over software defined networks. *Netw. Protoc. Algorithms* **2015**, *7*, 37–62.
27. Weigle, M.C.; Adurthi, P.; Hernández-Campos, F.; Jeffay, K.; Smith, F.D. Tmix: A Tool for Generating Realistic TCP Application Workloads in Ns-2. *SIGCOMM Comput. Commun. Rev.* **2006**, *36*, 65–76.
28. Botta, A.; Dainotti, A.; Pescapé, A. A tool for the generation of realistic network workload for emerging networking scenarios. *Comput. Netw.* **2012**, *56*, 3531–3547.

29. Rygielski, P.; Simko, V.; Sittner, F.; Aschenbrenner, D.; Kounev, S.; Schilling, K. Automated Extraction of Network Traffic Models Suitable for Performance Simulation. In Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering (ICPE '16), Delft, The Netherlands, 14 March 2016; pp. 27–35.
30. Smekal, D.; Hajny, J.; Martinasek, Z. Packet generators on field programmable gate array platform. In Proceedings of the 40th International Conference on Telecommunications and Signal Processing (TSP), Barcelona, Spain, 5–7 July 2017; pp. 97–100.
31. Muelas, D.; López de Vergara, J.E.; Berrendero, J.R.; Ramos, J.; Aracil, J. Facing Network Management Challenges with Functional Data Analysis: Techniques & Opportunities. *Mob. Netw. Appl.* **2017**, *22*, 1124–1136.
32. Muelas, D.; García-Dorado, J.; López de Vergara, J.E.; Aracil, J. Application of functional feature extraction to the compression of network time series. In Proceedings of the 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Lisbon, Portugal, 8–12 May 2017; pp. 592–595.
33. Mata, F.; García-Dorado, J.L.; Aracil, J. Detection of traffic changes in large-scale backbone networks: The case of the Spanish academic network. *Comput. Netw.* **2012**, *56*, 686–702.
34. Little, J.D. Little's Law as Viewed on Its 50th Anniversary. *Oper. Res.* **2011**, *59*, 536–549.
35. Rosenberg, J.; Schulzrinne, H.; Camarillo, G.; Johnston, A.; Peterson, J.; Sparks, R.; Handley, M.; Schooler, E. RFC 3261: SIP: Session Initiation Protocol. Available online: https://www.rfc-editor.org/rfc/rfc3261.txt (accessed on 23 February 2018).
36. Cisco Systems Inc. *Cisco Unified Communications Manager System Guide*; Chapter Understanding IP Telephony Protocols; Cisco Systems, Inc.: San Jose, CA, USA, 2004.
37. Jacobson, V.; Frederick, R.; Casner, S.; Schulzrinne, H. RFC 3550: RTP: A Transport Protocol for Real-Time Applications. Available online: https://tools.ietf.org/html/rfc3550 (accessed on 23 February 2018).
38. ITU-T. *G.711: Pulse Code Modulation (PCM) of Voice Frequencies*; International Telecommunication Union: Geneva, Switzerland, 1988.
39. García-Dorado, J.L.; Santiago del Río, P.M.; Ramos, J.; Muelas, D.; Moreno, V.; López de Vergara, J.E.; Aracil, J. Low-cost and high-performance: VoIP monitoring and full-data retention at multi-Gb/s rates using commodity hardware. *Int. J. Netw. Manag.* **2014**, *24*, 181–199.
40. Muelas, D.; López de Vergara, J.E.; Ramos, J.; García-Dorado, J.L.; Aracil, J. On the impact of TCP segmentation: Experience in VoIP monitoring. In Proceedings of the 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Lisbon, Portugal, 8–12 May 2017; pp. 708–713.