# Hardware-assisted virtual network benchmarking tools

Eric Rosche, Florian Wiedner*, Christoph Schwarzenberg*
*Chair of Network Architectures and Services
School of Computation, Information and Technology, Technical University of Munich, Germany
Email: eric.rosche@tum.de, wiedner@net.in.tum.de, schwarzenberg@net.in.tum.de

*Abstract*—**Accurate network emulation plays an essential role in research and development. Most of the time, only a couple of hosts are used to emulate comprehensive networks. This poses difficulties when focusing on singular network characteristics and their tail-behavior such as latency and throughput, as it is difficult to emulate all required aspects realistically. However, this is necessary, especially when looking at e.g. low-latency network emulation in aeronautics, where accurate measurements are crucial. In this paper, we examine how these realistic measurements are achieved by different hardware-assisted network emulators. We achieve this by comparing the different tools and drawing a conclusion from this analysis.**

*Index Terms*—**hardware-assisted, network benchmarking, network emulation, low-latency, high-throughput,**

## 1. Introduction

Network emulation is not a new concept. Common tools like Mininet [1] or NetEm [2] allow the emulation of configurable network topologies and their characteristics. Despite the existance of multiple tools that already allow such emulation, new tools and frameworks continue to be presented. Recent tools like Kollaps [3] and those by Sylla et al. [4], Ryu et al. [5], and Morin et al. [6] for wireless network emulation emphasize the ongoing relevance of this topic. New tools often focus on specific emulation features, such as low latency. When looking at low latency application, fields like Time-sensitive Networking and aeronautics, in which as-fast-as-possible and reliable calculations are crucial, emulation is a valuable tool. Other aspects like complexity and high throughput are interesting to emulate, as the importance of data centers with a huge amount of traffic over short links is only rising with further people and systems transferring to the cloud. These aspects are difficult to emulate only using virtual networking devices and require hardware assistance. This is made possible by specifications like Single-root Input/Output Virtualization (SR-IOV), which allow for Peripheral Component Interconnect Express (PCIe) to be virtualized.

In this paper, we will review multiple of these hardware-assisted tools, focusing on different network characteristics such as low latency and high throughput, and what design choices made this possible. In Section 2 Background about the topic will be provided. Section 3 presents the four different tools analyzed in this paper. Afterwards, we compare the tools in Section 4 and draw conclusions in Section 5.

## 2. Background and Related work

There are different types of network emulation tools. Some tools, like Mininet, are purely virtual solutions. These kind of tools are designed to emulate a full network on a single host and use features like namespaces for this purpose. Other tools can be defined as testbeds of multiple hosts, which are used on a large scale and by multiple people at once, spanning over one or multiple data centers. These testbeds are in their core real networks designed for emulating real traffic in a controllable environment. Testbeds like FABRIC [7], which was presented 2019, SCIONLab [8], which has been in use since 2016, or PlanetLab [9], established in 2003, are examples in this context.

In this paper, we will focus on a different kind of network benchmarking tool, namely the hardware-assisted variant. The tools analyzed in this paper all share the characteristic that they use real networking hardware to allow benchmarking, and are designed to only use a very limited number of devices. Depending on which hardware is used in the design, this allows for realistic data from real devices. We will also focus on tools which use off-the-shelf hardware.

### 2.1. Similar comparisons

Different comparisons of network emulation tools have been done before this analysis, though they focus on other aspects. In the article presenting the Kollabs emulator Gouveia et al. [3] present a very recent listing of available virtual network emulation tools. The main focus there lies in the comparisons of features, like dynamic changing of network properties, and in which manner the tools are implemented.

The article presenting the SCIONLab testbed [8] features a short table comparing different network testbeds on their features against the presented SCIONLab. We could not find a recent comparison of these large scale testbeds on a set of universal features. Comparisons like the one made by Mirkovic et. al in [10] from 2011 are older and therefore do not reflect the current iterations of these testbeds.

### 2.2. Simulators

In comparison to network emulation, network simulation only tries to replicate network behavior. Network simulators like OMNeT++ [11] and NS3 [12] therefore serve a different purpose. Similarly to the comparison of network emulators, we were not able to find recent network simulator comparisons. A comparison from 2014 by Kabir et. al [13] shows that with the huge amount of simulators available for different platforms and with different design goals, that a suitable simulator with the required features should be identifiable.

## 2.3. SR-IOV

SR-IOV is a standard which allows a single PCIe device to be split into multiple virtual devices, which can then be used in virtualization. To achieve this, SR-IOV uses virtual and physical functions. Physical function are complete PICe functions, which allow for configuration of the device in the standard way. Virtual Functions (VF) on the other hand are soly capable of sending and receiving data.

This allows for different Virtual Machines (VM) to use the same PCIe device without any other kind of resource allocation. The hypervisor or the managing operating system of the VMs must support SR-IOV, as the physical function could be required at some point in the virtualization process. This can speed up the virtualization of e.g. networking devices as are used by the tools discussed in this paper [14].

## 3. Hardware-Assisted Virtual Network Emulation Tools

In the following, we will introduce the tools we will be comparing. All of the following tools provide the ability to benchmark virtual network configurations while utilizing hardware assistance. We have selected these tools for comparison because they employ similar techniques, such as SR-IOV, but they have different approaches and design goals.

### 3.1. HVNet

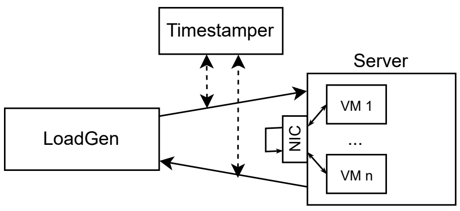

Figure 1: Simplified HVNnet setup

HVNet [15] is an approach for creating virtual network topologies utilizing real networking hardware, with a focus on realistic timestamping of low-latency traffic. To achieve this, a configuration with a load generator that generates traffic, connected to a Device under Test (DuT) using HVNet is used. For the evaluation, timestamping hardware is used to ensure a minimal discrepency. The traffic sent and received by the DuT is channeled over a Network Interface Card (NIC) utilizing SR-IOV to separate it into multiple virtual links. VMs running on the DuT are configured to have minimal overhead. To achieve a setup with VMs instead of using the Data Plane Development Kit (DPDK) [16], like e.g. Mininet is using, a kernel-based networking approach is employed. This setup allows for extremely low-latency traffic emulation, where the 99th percentile of logical link latency is approximately 100 µs for 2-hop measurements at 1 Mbit/s. This tool aims for realistic network behavior, emulating low latency, low jitter network traffic on a easily configurable setup with minimal devices. HVNet setup is summarized in Figure 1.

### 3.2. NFV-TestPerf

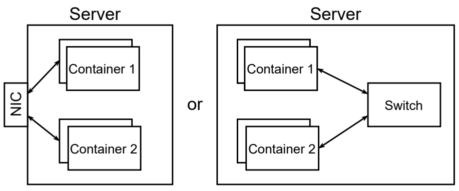

Figure 2: Simplified NFV-TestPerf single host setup

Another way to emulate multiple hosts on a single computer is by using containers. Similar to VMs, containers can be assigned a virtual function on an SR-IOV capable network card. A tool that allows this is NFV-TestPerf [17]. This framework can be used to specify network topologies and configure applications hosted in containers to communicate over different connection types like Linux bridges, virtual switches, or the aforementioned SR-IOV-capable networking hardware. We will focus on the usage of SR-IOV and also discuss the results using VALE [18], a virtual networking switch.

Another feature achieved by this approach, is the framework being very flexible. It allows for single and multiple container communication on single or multiple host configurations, each with different connection types as described above. Six different connection types are compared in multiple scenarios with single or multiple hosts. To achieve network virtualization, DPDK's Application Programming Interface (API) is used. It allows the virtual containers to access the VFs of the SR-IOV capable hardware as well as the virtual software switches. Therefore, the network virtualization runs in user space.

Using this method, tests performed on only a single host with a low packet sending rate and a maximum burst size of 128 achieved latencies below 200 µs using SR-IOV. The NFV-Testperf is summarized in Figure 2.

### 3.3. TurboNet

A different approach to emulating network behavior with the help of networking hardware is TurboNet [19]. The idea here is to use a programmable switch to emulate multiple configurable virtual switches. The main goal to TurboNet is, in contrast to the two tools mentioned earlier, to emulate switching topologies. To achieve this, the programmable switch is sliced into multiple emulated switches and links. Packets are sent over physical links only when they enter or exit the switch topology.

This is accomplished by assigning each virtual switch a set of ports and emulating link delays via a queue. Additionally, TurboNet allows for multiple programmable switches to be used by connecting them via a physical link.

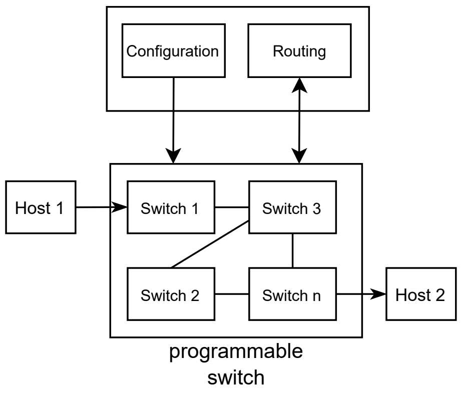

Figure 3: Simplified TurboNet setup

Configuration of TurboNet is available via an API, which also allows for configuring link behaviors such as loss and artificial delays. The TurboNet setup is summarized in Figure 3.

## 3.4. ExRec

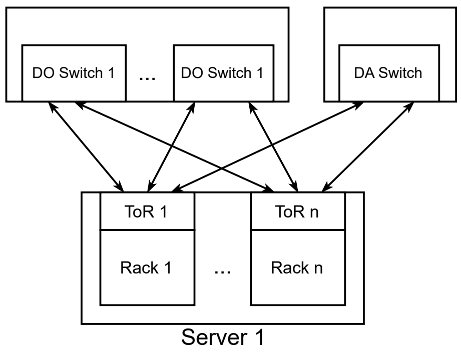

Figure 4: Simplified ExRec single host setup

As data centers become increasingly relevant and continue to grow in size, they represent another crucial application for network emulation. An example of a framework tackling this issue with off-the-shelf hardware is ExRec [20]. In the setup of ExRec, a variable number of hosts M emulates N data center racks, which are connected to top-of-rack switches emulated on the M hosts with virtual machines. The ToRs are then connected to an electric packet switch, which emulates k demand-oblivious spine switches. An optical circuit switch is used as a demand-aware switch. The framework also ensures that network bottlenecks occur only as intended, and that there are no bottlenecks on the VMs. For control messages, the most achievable inter-arrival time was 500 µs, but the focus of the testbed is on high throughput with realistic data center behavior. The ExRec setup is summarized in Figure 4.

## 4. Comparison of the Tools

In the following, we will try to compare the designs, features, and limitations of the previously presented tools. We will focus on the purpose each tool serves and how they achieve this in comparison to the other tools. The comparison is summarized in Table 1.

### 4.1. Low Latency

Three of the four mentioned tools have low latency as a requirement or feature. HVNet is specifically designed with low latency in mind, as is NFV-TestPerf. In both cases, SR-IOV capable network cards can be used to transfer packets between different virtual network nodes.

TurboNet uses emulated switches, which only have a nanosecond delay because of the use of loopback ports as links. In the evaluation, a Tofino switch is used for which the dequeuing plus enqueuing delay adds up to around 200 ns for all available bandwidths from 10 Gbit/s to 100 Gbit/s. As this is not a realistic representation of larger networks, TurboNet uses queue depth in programmable switches to bridge this delay to a more realistic value. Even with passing a 10 Gbit/s delayed queue, the delay ranges from 100 µs to 1000 µs.

ExRec does not have a focus on low latency directly. Latency measurements are not directly highlighted in the article describing the tool, but in the evaluation of the testbed, switches are configured at runtime, which is displayed to be possible with an inter-arrival time of about 500 µs. This, however, only gives us a very low bound of what the actual worst case latency of packets at full load could be, which should be significantly higher. For latency, however, only the worst case is an interesting metric, as a big deviation can lead to unreliable results. In this regard, it is, therefore, limited in comparison to the other tools.

When comparing HVNet and NFV-TestPerf, HVNnet with its $\approx$ 200 µs worst-case latency for 2 hops, and NFV-Testperf with a mean latency between 100 µs to 200 µs for a burst rate of 128 and 1500-byte packets, seem to both reach a good performance. It is, however, important to note that virtual networking for HVNet is done in the kernel space, where NFV-Testperf's is done in user space. Additionally, NFV-Testperf uses packets with a size of 1500 bytes, where HVNet uses packets with a size of 363 bytes. But when pulling Mininet as a fully virtual alternative to the test, as has been done in the publication presenting HVNet [15] it has worst-case latencies of about 600x (about 100 ms). This shows that to achieve these kinds of latencies, optimized processes and hardware are absolutely necessary.

### 4.2. Throughput

Another very interesting network metric is throughput. Especially when emulating multiple virtual hosts on a single real one, high throughput can be difficult to achieve when using real networking hardware. Using an SR-IOV capable network card for multiple virtual links, the maximal achievable throughput for each virtual link is the maximal throughput the network card can handle divided by the amount of virtual links [15]. In more complex emulated topologies, this can quickly become an issue. Looking at a 10 Gbit/s network card, only 10 virtual links can be assigned to physically reach a Gbit/s one-directional link throughput. Additionally, packet size can also have a big influence on throughput, as reducing the packet size by half doubles the cost for the same throughput. The second limit is, therefore, the emulation cost coming from high throughput traffic. This can reduce the emulatable network size in the worst case exponentially

TABLE 1: Comparison of Tools

| Tool | Topology | Networking Hardware | Latency | Throughput |
|---|---|---|---|---|
| Mininet | fully virtual hosts | - | ≈ 200 ms | 1 Gbit/s |
| HVNnet | LoadGen, Timestamper, and DuT | SR-IOV NIC | worst-case ≈ 200 µs | N/A |
| NFV-TestPerf | Host spawning containers | u.a. SR-IOV NIC | mean ≈ 100 µs to 200 µs | ≈ 13 Gbit/s |
| TurboNet | Emulated switches in programmable switch | programmable switch | emulated ≈ 200 µs to 1000 µs | 40 Gbit/s |
| ExRec | hosts virtualizing racks connected to switches | NIC and emulated switches | min 500 µs | 10 Gbit/s |

It is important to add that these tools were evaluated in their presenting article and are therefore using different hardware. The displayed values are to be viewed as a reference. Mininet throughput from [21]

if every virtual host would be connected to every other virtual host.

TurboNet tackles this issue by implementing its own background traffic emulation. This is done by using programmable switches to inject packets into the switch pipeline. In a comparison made by Emmerich et al. in the paper presenting MoonGen [22], a popular packet creation tool, it is struggling to reach a comparable kind of throughput for smaller packet sizes.

ExRec is another tool that focuses more on achieving throughput goals, despite the emulation of switches. In their evaluation, the tool could reach high throughput levels extremely fast, right after a certain preconfigured flow had been started.

As NFV-TestPerf's primary goal is to emulate virtualized network functions, which can also need high volumes of data depending on the application, throughput is also a metric interesting here. SR-IOV has some of the best achieved results in the throughput evaluation of the tool, only being beaten by VALE in some scenarios, where throughput reaches up to 12 Gbit/s.

## 4.3. Results

After comparing the different tools, we can draw some results and requirements for this kind of tools.

**4.3.1. Low Latency.** For measuring low latency traffic, hardware supporting virtualization appears necessary though measurements using VALE returned good results in the evaluation of NFV-TestPerf [17]. This could be interesting to analyze further for the other tools, especially HVNet as it has a similar structure. In comparison, all the tools achieve latencies that are far below what Mininet and other purely virtual solutions can achieve.

**4.3.2. High Throughput.** When looking at throughput, a different conclusion can be drawn. As throughput is limited by the hardware, using VALE achieved good results in the evaluation of NFV-TestPerf [17]. The tests where this was conducted were limited to a very small number of containers. It is expected that for more virtualized nodes this result will only strengthen, as then the worst-case exponential limitation of the hardware will come into effect. It would however be interesting to analyze the load that is then put on the CPU, as it would rise in the same manner as the hardware limitation. Overhead from each host also plays a role here, as it doesn't limit the configuration using hardware, but the software switching may be affected. This is why NFV-TestPerf is implemented to only allow network bottlenecks.

**4.3.3. Final Points.** The comparison with Mininet shows that specialized setups can achieve realistic low latency and also high throughput traffic even when simulating bigger network topologies with only very few hosts. But other than Mininet, which is designed to be as flexible as possible, a lot of additional work is required to allow for more emulation configurations. Tools like ExRec and TurboNet can only emulate a very specific setup and cannot be used to emulate a more general network topology. HVNet and NFV-TestPerf are designed to allow a more general topology but may face challenges when simulating more specific network types, e.g., switching, like TurboNet is designed to, or data center behavior as is done by ExRec. It, therefore, becomes clear that a tool for all purposes, like Mininet, cannot create the best results for specific network aspects, and that this focus requires specialized setups in itself. So if an emulator is to be chosen for a project and the standard tools do not meet the requirements for the emulation, an emulator specialized in the characteristics of the project may be hard to find. Some of the tools that have been compared in this article are available open source, namely ExRec and NFV-Testperf, and the links to GitHub are still active and working at the time of writing.

## 5. Conclusion and Future Work

In this paper, we compared different hardware-assisted network emulation tools for their features. This allowed us to verify the importance of hardware in emulators where low latency is a design goal. We also learned that focusing on a few characteristics of networks can easily lead to restrictions concerning other characteristics. When using NICs in the emulation process, the number of links that are emulated over it imposes a harsh restriction on the achievable throughput.

However, if no hardware is used, the tools have significant downsides in these regards.

The comparison leaves a few open questions that could be answered in future work. For example, the comparison of the tools was done with a focus on the emulation of low latency networks and throughput. Other, more abstract characteristics of networks could be compared, like the emulation of a network with a high number of nodes or different scenarios like cross-traffic load. Additionally, the tools could be compared in a more practical way by setting them up and running tests on them. Other aspects mentioned in Section 4.3 could also be compared, like the performance of VALE in the other tools.

## References

[1] "An Instant Virtual Network on your Laptop (or other PC)," mininet.org, [Online; accessed 14-July-2023].

[2] S. Hemminger, "Network Emulation with NetEm," April 2005.

[3] P. Gouveia, J. a. Neves, C. Segarra, L. Liechti, S. Issa, V. Schiavoni, and M. Matos, "Kollaps: Decentralized and dynamic topology emulation," 2020. [Online]. Available: https://doi.org/10.1145/3342195.3387540

[4] T. Sylla, L. Mendiboure, M. Berbineau, R. Singh, J. Soler, and M. S. Berger, "Emu5gnet: an open-source emulator for 5g software-defined networks," in *2022 18th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2022, pp. 474–477.

[5] B. Ryu, R. Knopp, M. Elkadi, D. Kim, and A. Le, "5g-emane: Scalable open-source real-time 5g new radio network emulator with emane," in *MILCOM 2022 - 2022 IEEE Military Communications Conference (MILCOM)*, 2022, pp. 553–558.

[6] D. G. Morin, P. P. ManuelJ. López Morales, and A. G. A. A. Villegas, "FikoRE: 5G and Beyond RAN Emulator for Application Level Experimentation and Prototyping," 2022.

[7] I. Baldin, A. Nikolich, J. Griffioen, I. I. S. Monga, K.-C. Wang, T. Lehman, and P. Ruth, "Fabric: A national-scale programmable experimental network infrastructure," *IEEE Internet Computing*, vol. 23, no. 6, pp. 38–47, 2019.

[8] J. Kwon, J. A. García-Pardo, M. Legner, F. Wirz, M. Frei, D. Hausheer, and A. Perrig, "Scionlab: A next-generation internet testbed," in *2020 IEEE 28th International Conference on Network Protocols (ICNP)*. IEEE, 2020, pp. 1–12.

[9] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "Planetlab: an overlay testbed for broad-coverage services," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 3–12, 2003.

[10] J. Mirkovic, A. Hussain, and H. Shi, "A comparative study of network testbed usage characteristics."

[11] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment," in *1st International ICST Conference on Simulation Tools and Techniques for Communications, Networks and Systems*, 2010.

[12] "ns-3 Network Simulator," https://www.nsnam.org, [Online; accessed 14-July-2023].

[13] M. H. Kabir, M. J. H. Syful Islam, and S. Hossain, "Detail comparison of network simulators," vol. 5, no. 20, 2019.

[14] P. Legros, "Why using Single Root I/O Virtualization (SR-IOV) can help improve I/O performance and Reduce Costs," https://www.design-reuse.com/articles/32998/single-root-i-o-virtualization.html, [Online; accessed 02-August-2023].

[15] F. Wiedner, M. Helm, S. Gallenmüller, and G. Carle, "Hvnet: Hardware-assisted virtual networking on a single physical host," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2022, pp. 1–6.

[16] "Data Plane Development Kit," https://www.dpdk.org/, [Online; accessed 02-August-2023].

[17] G. Ara, L. Lai, T. Cucinotta, L. Abeni, and C. Vitucci, "A framework for comparative evaluation of high-performance virtualized networking mechanisms," in *Cloud Computing and Services Science*, D. Ferguson, C. Pahl, and M. Helfert, Eds. Cham: Springer International Publishing, 2021, pp. 59–83.

[18] L. Rizzo and G. Lettieri, "Vale, a switched ethernet for virtual machines," in *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, 2012, pp. 61–72.

[19] J. Cao, Y. Liu, Y. Zhou, L. He, and M. Xu, "Turbonet: Faithfully emulating networks with programmable switches," *IEEE/ACM Transactions on Networking*, vol. 30, no. 3, pp. 1395–1409, 2022.

[20] J. Zerwas, C. Avin, S. Schmid, and A. Blenk, "Exrec: Experimental framework for reconfigurable networks based on off-the-shelf hardware," in *Proceedings of the Symposium on Architectures for Networking and Communications Systems*, 2021, pp. 66–72.

[21] A. Al-Sadi, A. Al-Sherbaz, J. Xue, and S. Turner, *Developing an Asynchronous Technique to Evaluate the Performance of SDN HP Aruba Switch and OVS: Proceedings of the 2018 Computing Conference, Volume 2*, 01 2019, pp. 569–580.

[22] P. Emmerich, S. Gallenmüller, D. Raumer, F. Wohlfart, and G. Carle, "Moongen: A scriptable high-speed packet generator," in *Proceedings of the 2015 Internet Measurement Conference*, 2015, pp. 275–287.