Latency Measurement as a Virtualized Network Function using Metherxis*

Diego R. Mafioletti
Alextian B. Liberatto
IFES - Colatina/ES
diego.rossi@ifes.edu.br
alextian@ifes.edu.br

Cristina K. Dominicini
IFES - Serra/ES
cristina.dominicini@ifes.edu.br

Rodolfo S. Villaca
Magnos Martinello
Moises R. N. Ribeiro
UFES - Vitoria/ES
rodolfo.villaca@ufes.br,
magnos@inf.ufes.br,
moises@ele.ufes.br

ABSTRACT

Network latency is critical to the success of many high-speed, low-latency applications. RFC 2544 discusses and defines a set of tests that can be used to describe the performance characteristics of a network device. However, most of the available measurement tools cannot perform all the tests as described in this document. As a novel approach, this paper proposes Metherxis, a tool that can be used on general purpose hardware and enables Virtualized Network Functions (VNFs) to measure network device latency with micro-second grade accuracy. Results show that Metherxis achieves highly accurate latency measurements when compared to OFLOPS, a well known measurement tool.

CCS Concepts

•Networks → Network measurement;

1. INTRODUCTION

Network latency measurements are crucial in providing reliable and efficient networked services, such as e-commerce, multimedia streaming, and social networking. Most of these service providers run their servers on clouds, which are geographically distributed and far away from their users. Furthermore, highly accurate latency measurements are increasingly critical to the success of many high-speed, low-latency applications, such as trading transactions, databases that require improved timestamp accuracy, and server synchronization for automation or regulatory purposes.

A latency measurement is composed of three main components: propagation, transmission and processing delay. The first two components depend only on the distance, the physical media and the network bandwidth. Processing delay, in turn, depends on the computing system, which can be a server, a desktop, a smartphone or a network device. In terms of a network device latency measurement, the reference is RFC 2544 [1], which defines a set of tests that can be used to describe performance characteristics.

In the literature, there are several tools to evaluate the performance of network devices; among them, we highlight `pktgen` [3], a software tool, and `OFLOPS` [2], a NetFPGA based platform. All the related tools suffer from at least one of the following limitations: do not provide the level of accuracy, flexibility and scalability required by high performance applications; cannot perform all the tests described in RFC 2544 due to software or hardware limitations; or require high investment on specialized hardware.

To address these challenges, Metherxis is presented in Section 2 as a novel approach to measure latency with micro-second accuracy as Virtualized Network Functions (VNFs). Metherxis is publicly available[1] and can be deployed on standard hardware and hosted in a cloud infrastructure. The virtualization technique implemented by LinuX Containers (LXC) allows Metherxis to be vertically scalable according to the number of network interfaces in the physical host and to deploy containers that share the same clock in order to measure the latency with high accuracy.

In Section 3, a loopback mode procedure is performed to compare Metherxis and `OFLOPS`. Moreover, this section evaluates the behavior of the selected tools under packet size and packet rate variations. Section 4 concludes the paper and points to future work.

2. METHERXIS

The key idea of Metherxis is to employ a single Linux host to allow the creation of a wide range of Virtualized Network Measurement Functions (VNMFs). To accomplish this task, it relies on LXC for resource isolation.

A VNMF in Metherxis is composed of two building blocks: a packet generator and a packet analyzer. The packet generator represents a sender (TX) and the packet analyzer represents a receiver (RX). Depending on the required measurement setup, one VNMF may require multiple senders (TX) and/or receivers (RX).

As shown in Fig 1, each building block has its own physical network interface and it is assigned to a specific namespace container. In this way, the building blocks can be vertically scalable from 1 to n. For measuring the latency with high accuracy, a packet generator can be deployed in namespace 0 (TX) generating traffic at physical ethernet port 0 and a packet analyzer can be set to receive packets in a namespace 1 (RX) at port 1.

Metherxis Implementation: We chose `pktgen` [2] to generate packets in kernel mode. The latency measurement using `pktgen` requires the addition of a timestamp to the

*This paper was originally published in the proceedings of the LANCOMM 2016 workshop.

[1]http://git.lprm.inf.ufes.br/diego/metherxis
[2]https://pktgen.readthedocs.org/en/latest/

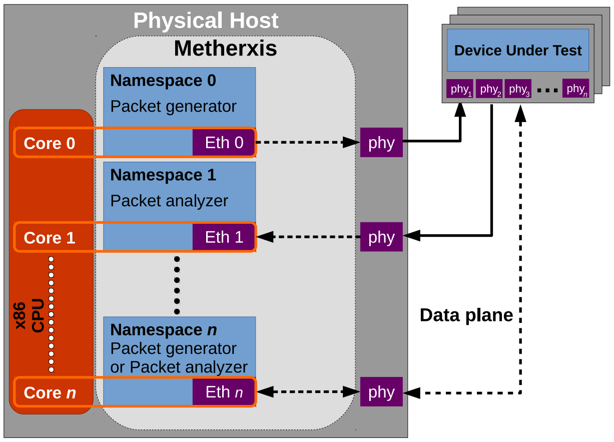

Figure 1: Metherxis conceptual view.

packet at its departure time. When the packet arrives at the receiver side, the receiver subtracts the current system clock from the marked timestamp. The result of this process is the packet latency measurement and its accuracy depends on how fast the packets are processed.

However, `pktgen` is designed for a single host and is not optimized for the LXC concept. Thus, in order to enable the communication between the containers, Metherxis modifies `pktgen` and implements, in kernel mode, one timestamp array for each sender, which can be read by any receiver. Each array is uniquely identified by the sender source IP address and stores one tuple $< key, value >$ for each packet, which contains the packet identification and its timestamp, respectively. It is inserted in the Identification field of the IP protocol. Our assumption for this implementation is that there is no IP fragmentation between the sender and the receiver involved in a given measure.

The timestamp is set into the array when the packet leaves the kernel to the network interface. At the receiver (RX), Metherxis reads the system clock at the packet arrival time subtracting it by the timestamp value stored at the position key of the array. In contrast to `pktgen` that adds timestamps in the packet, Metherxis does not modify the payload, reducing the packet processing time and, consequently, increasing the measurement accuracy. The best accuracy is guaranteed by Metherxis when each CPU core processes IRQs of a specific network interface.

The implementation is scalable since only the sender writes in its own array and any receiver is able to read timestamp values from sender's array.

3. BENCHMARKING METHERXIS

This section aims at evaluating the accuracy tradeoffs provided by different measurement approaches: OFLOPS and Metherxis. In order to define a widely known set of tests, our benchmarks follow RFC 2544. Each experiment was repeated 30 times and we plot the average results with a 95% confidence interval. For the loopback tests, we created a physical loopback by connecting the sender port to the receiver port.

As the first step, we evaluate three 1 Gbps NIC models by performing the tests defined in RFC 2544 with packet size of 64 Bytes. The goal of this test was to evaluate the NIC bottlenecks. Figs. 2 and 3 compare the number of dropped packets and latency values when varying packet rates from 100 to 1000 kpps. Beyond 1000 kpps all tested NICs are no longer loss free.

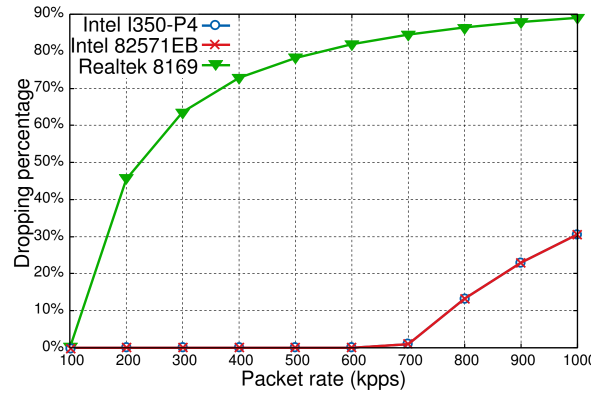

Figure 2: Dropping Packets Evaluation of Network Interface Cards

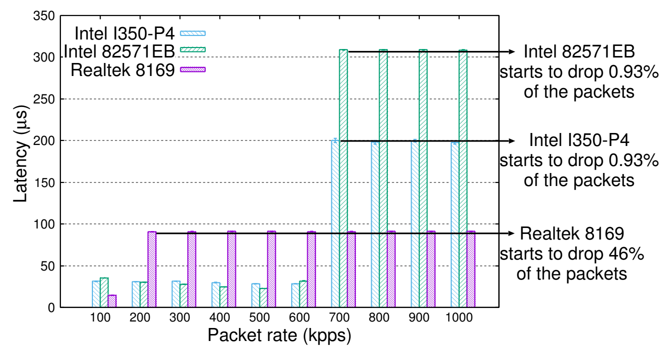

Figure 3: Latency Evaluation of Network Interface Cards

In Fig 2 both Intel NICs reached the same results. Fig. 3 presents latency assessment at micro-second grade for the successfully transmitted packets. Note that latency tends to be insensitive to packet rate up to the point when packet losses start to happen. After that, a higher latency plateau is reached with severe packet loss. Intel I350-P4 was the last NIC to reach the second plateau. Thus, this NIC was selected to the remaining evaluations.

The next benchmark is a comparison with OFLOPS[2]. The goal is to evaluate the percentage of packets sampled (time-stamped) for latency calculation in both tools. The results show that the way packets are time-stamped in OFLOPS creates a severe bottleneck for latency measurements.

Fig. 4 shows a comparison between OFLOPS and Metherxis in loopback mode. It is clear that the OFLOPS approach quickly limits measurement statistics by severely reducing the percentage of time-stamped packets as packet lengths decrease. Even with the standard MTU size, less than 30% of transmitted packets are used for latency calculations. OFLOPS limitation is critical at 64 Bytes packets as no packets are used for latency computation. As a result, OFLOPS cannot be used for RFC 2544 compliant tests. On the other hand, the lightweight system adopted by Metherxis at kernel level

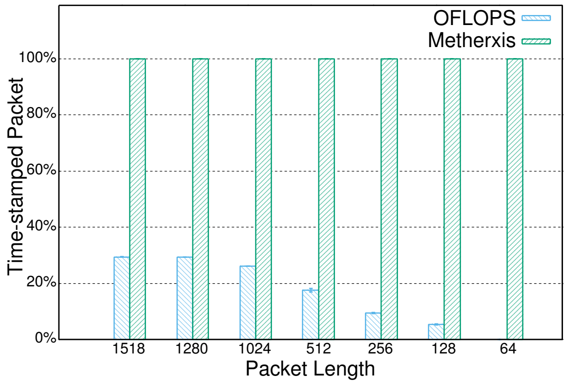

Figure 4: Percentage of time-stamped packets for variable packet size in loopback

not only generates and receives packets at wire speed, but it also considers 100% of the packets.

In order to further investigate OFLOPS' limitations, another test was performed by setting packet length at the shortest RFC 2544 packet size that OFLOPS can support (128 Bytes). Then, we selected different packet rates in order to evaluate its effect on the percentage of time-stamped packets. As seen in Fig. 5, in OFLOPS, latency is computed with a very few useful samples as the packet rate increases.

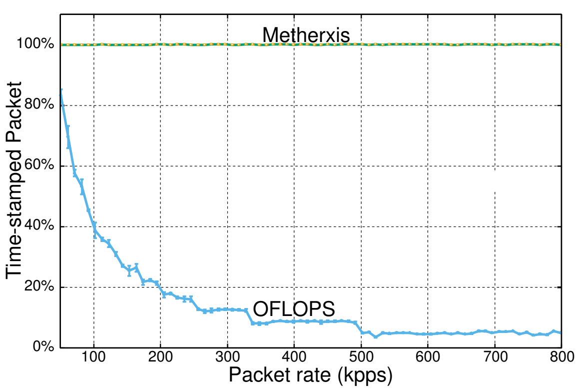

Figure 5: Percentage of time-stamped packets for variable packet rate in loopback

Fig. 5 clearly shows how packet rate affects OFLOPS framework as it barely reach 20% time-stamped packets at 200 kpps. In contrast, Metherxis is able to support 100% time-stamping with different packet rates.

More importantly are the latency measurements in loopback mode, as they set the tool baseline accuracy. To this end, Fig. 6 presents OFLOPS and Metherxis latency measurements for 128 Bytes packet size, as packet rates increase. As expected, OFLOPS with its dedicated hardware is not able to measure latency in loopback for packet rates below 500 kpps (bear in mind that OFLOPS uses SNMP timestamping, for which the granularity is 1 μs). Nevertheless, for packet rates above 500 kpps, its loopback latency jumps to a 38 μs plateau. On the other hand, Metherxis can reach stable and well-behaved loopback latency at different packet rates. Using off-the-shelf NICs, Metherxis reaches between 32 and 36 μs for 200 kpps and beyond.

It is worth mentioning that, despite its variation for lower packet rates, the 95% confidence interval bar is not visible for the whole range. This fact allows us to calibrate Metherxis for any packet rate, since the average values are reliable references that can be later subtracted from the actual measurements taken from the DUT (Device Under Test). As a result, Metherxis enables inexpensive micro-second grade measurement tools to be built using its basic blocks.

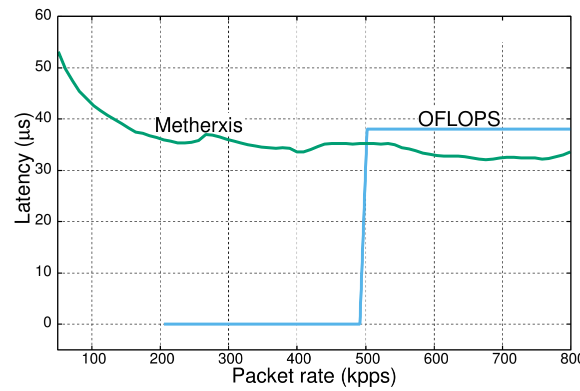

Figure 6: Metherxis x OFLOPS latency measurements for 128 Bytes packet and variable packet rate

4. CONCLUSION AND FUTURE WORK

This paper presented a new concept for enabling multi-port micro-second grade latency measurements from inexpensive o-the-shelf x86 hardware with comparable performance to other solutions based on specialized hardware such as NetFPGA. This was only possible thanks to the new concept of lightweight virtualization in Linux containers. Physical loopback measurements were used to benchmark the system in comparison to OFLOPS. This opens new avenues for latency measurements and scalable virtualized measurement network functions, whereas complex measurements can be created out of Metherxis building blocks.

5. REFERENCES

[1] Bradner, S., and McQuaid, J. Rfc2544: Benchmarking methodology for network interconnect devices, 1999.

[2] Rotsos, C., Sarrar, N., Uhlig, S., Sherwood, R., and Moore, A. W. Oflops: An open framework for openflow switch evaluation. In *13th International Conference on Passive and Active Measurement* (Berlin, Heidelberg, 2012), PAM'12, Springer-Verlag, pp. 85–95.

[3] Torrents, D. T. Open Source Traffic Analyzer. Master's thesis, KTH Information and Communication Technology, Stockholm, Sweden, 8 2010.

Acknowledgments

This work has received funding from CNPq, CAPES and FAPES and it is part of the FUTEBOL project, which has received funding from the European Union's Horizon 2020 under grant agreement no. 688941 (FUTEBOL), as well from the Brazilian Ministry of Science, Technology and Innovation (MCTI).