Network Emulation Support in ns-3 Through Kernel Bypass Techniques

Pasquale Imputato
Università degli studi di Napoli "Federico II"
pasquale.imputato@unina.it

Stefano Avallone
Università degli studi di Napoli "Federico II"
stefano.avallone@unina.it

Tommaso Pecorella
Università degli studi di Firenze
tommaso.pecorella@unifi.it

ABSTRACT

The network emulation support in ns-3 enables the interaction with real devices allowing the integration of a simulated node or network in an emulated scenario. Also, the network emulation support in a simulator is fundamental to validate its models. For both purposes, the emulation capabilities in ns-3 must reproduce the real host incoming and outgoing networking paths to increase the emulation credibility. In this work, we present our effort to introduce kernel bypass techniques in ns-3 to improve the network emulation support. We used netmap, a fast packets processing framework, to bypass the host networking stack and gain direct access to network device. We designed and introduced in ns-3 a new device to perform emulation that uses the netmap primitives to read and write packets on a real device. We carried out a preliminary benchmark of our device in an ethernet back-to-back scenario. The results show a more accurate performance in terms of delay and more realistic distributions of packets backlog in traffic-control of ns-3 and of packets inflight.

CCS CONCEPTS

• **Networks** → **Network simulations**; **Network experimentation**; • **Hardware** → **Simulation and emulation**;

KEYWORDS

ns-3, Traffic Control, Queueing Discipline, Emulation, netmap

ACM Reference format:
Pasquale Imputato, Stefano Avallone, and Tommaso Pecorella. 2017. Network Emulation Support in ns-3 Through Kernel Bypass Techniques. In *Proceedings of 11th EAI International Conference on Performance Evaluation Methodologies and Tools, Venice, Italy, December 5–7, 2017 (VALUETOOLS 2017),* 2 pages.
https://doi.org/10.1145/3150928.3150966

1 INTRODUCTION

The network emulation support in ns-3 enables the interaction of the simulator with real devices. This interaction is required to integrate a simulated node or network in a emulated scenario. Also, in network simulation, the network emulation support is used to model validation.

VALUETOOLS 2017, December 5–7, 2017, Venice, Italy

ACM ISBN 978-1-4503-6346-4/17/12.
https://doi.org/10.1145/3150928.3150966

ns-3 supports network emulation by standard communication primitives, i.e., a raw socket, to interact with the real devices [1]. Therefore, the emulation in ns-3 takes the effect of the host networking stack. This work introduces a technique of kernel bypass to improve the emulation capabilities on real devices in ns-3. We used netmap [4], an high performance packets processing framework, to bypass the host networking stack and gain direct access to the device rings. We introduced a new device in ns-3, which recreate more realistic incoming and outgoing networking paths, that use the netmap primitives to read and write packets on real devices. Also, we designed and implemented a flow control mechanism between the netmap ring and the simulated traffic-control and device support for Byte Queue Limits (BQL).

The new device allows faithful reproduction of networking paths in scenarios of real devices emulation in ns-3. Such behavior is fundamental in order to perform model validation and to assess upper layers performances such as the effectiveness of the Active Queue Managements Algorithms (AQM) or of the Delay Tolerant Network (DTN) protocols.

The remainder of this paper is organized as follows. Section 2 presents some background about ns-3 in emulated scenarios. Section 3 describes the device model and its implementation. Section 4 shows some preliminary results. Section 5 concludes our work.

2 BACKGROUND

ns-3 supports emulation by the FdNetDevice that uses a raw socket to interact with the emulated device. Basically, all the queues are outside ns-3 and no queue is built in ns-3 traffic-control. Therefore, the FdNetDevice in emulation mode takes the effects of the host networking stack in terms of host traffic-control and NIC queueing. Also, there is no flow control mechanism between ns-3 and the raw socket.

The ns-3 emulation performances are discussed in [1] with moderate traffic load. Basically, all the packets backlog is outside ns-3. The delay performance is conditioned by this behavior and no drops occur. In ns-3 has been introduced a realistic traffic differentiation infrastructure modeled after the Linux one [2, 3]. Since there is no backlog in ns-3 in emulation scenarios, the user cannot explore the effect of packets manipulation in the simulated queueing disciplines, e.g., by adding random delay, or the effect of the AQM algorithms on the packets backlog. Finally, the simulator cannot inspect the NIC ring to implement a realistic flow control. A flow control mechanism is requested to recreate a realistic outgoing path. Indeed, the device implemented in ns-3 must stop its queue when there is no room for other packets and reclaim other packets when a slot becomes available.

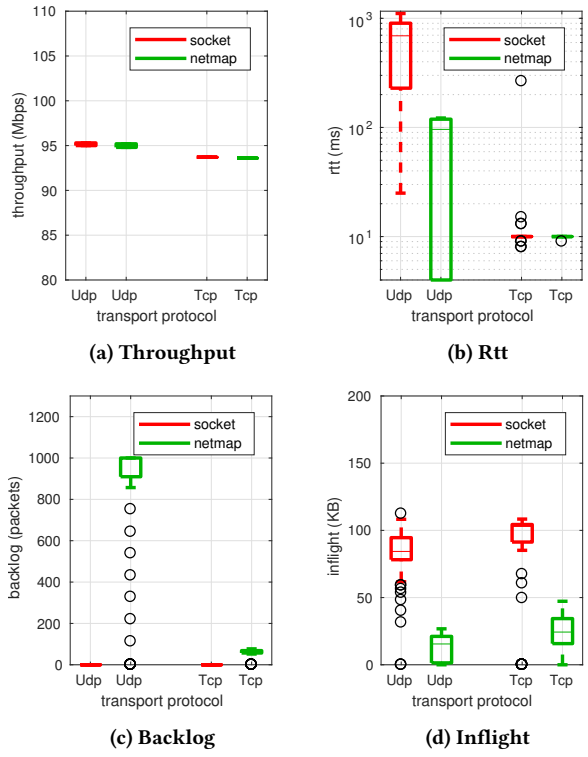

Figure 1: Evaluation of the impact of the new device on network performance.

3 DESIGN AND IMPLEMENTATION

We implemented a new ns-3 netdevice, named NetmapNetDevice, which inherits from the existing FdNetDevice and specializes the initialization, as the real device has to be put in netmap mode, and the send/receive methods, which have to make use of the netmap API to exchange packets with the netmap ring. The Send method, which is the one called by the traffic-control module to pass a packet to the netdevice, uses the netmap API to write the packet to a slot available in the netmap ring. Afterwards, the Send method checks whether there is enough room in the netmap ring for another packet. If not, the NetDeviceQueue object associated with the netmap ring is stopped. If netmap is operating in native mode, i.e., the netmap ring correspond to the NIC ring, it means that also the NIC ring does not have enough room for another packet. Starting a NetDeviceQueue object that has been stopped is a bit tricky, given that the netmap API does not notify the applications when some slots are made available in the netmap ring. The solution we implemented consisted in a separate thread that is signaled of the queue stopped event and poll for the next slot available in the netmap ring. The NetmapNetDevice we implemented also supports BQL. To this end, the Send method also notifies the BQL library of the amount of bytes that have been written to the netmap ring. The notification to BQL of the amount of bytes that have been transferred to the NIC exploits the thread to periodically notify about the transmitted bytes.

It is worth noticing that this device can operate with netmap in both native and emulated (or generic) mode. netmap can operate in native mode with a netmap-aware device driver. In such case, the netmap ring corresponds to the NIC ring. Conversely, in emulated mode netmap works without device driver changes. In emulated mode netmap insert the packets along the driver path.

4 RESULTS

We carried out a preliminary benchmark of the new device in a back-to-back 100 Mbps ethernet link of two hosts equipped with Intel e1000e ethernet card. The sender host uses a simulated generator and offers 100 Mbps of UDP or TCP traffic with packets size of 1400 bytes through netmap or socket depending on the emulation mode. We used netmap in emulated mode with generic_txqdisc disabled and BQL enabled. We reported in Figure 1 the received throughput, the ping RTT, the packets backlog in the ns-3 traffic-control and the bytes inflight outside the ns-3 simulator waiting for transmission. We note that the netmap outgoing path has bytes inflight within the netmap ring and in the NIC ring (when netmap works in emulated mode) while the socket path has bytes inflight in the host traffic-control and in the NIC ring.

The new device received throughput is very similar to the socket one. The RTT performance is reduced with the new device of a factor of 7 in the median in case of UDP flow and it is very similar in case of TCP flow. The new device, with flow control and BQL enabled, leads to a packets redistribution. Indeed, the ns-3 traffic-control backlog is not null, which is extremely important to evaluate traffic control techniques. Moreover, the bytes inflights are reduced, demonstrating the effectiveness of BQL technique.

5 CONCLUSIONS

This work introduces a technique of kernel bypass to improve the emulation capabilities of ns-3. The new device uses the netmap primitives to bypass the host networking stack and gain direct access to the device rings. Our preliminary evaluation shows that the new device leads to an improvement in the network parameters and a more realistic packets distributions of the packets waiting for transmission inside the simulator and outside the simulator. We believe that our device will allow to explore more realistic emulation scenarios with simulated components and to improve the model validation approach.

ACKNOWLEDGMENTS

This work is partially supported by the ESA Summer of Code in Space 2017 with ns-3.

REFERENCES

[1] Gustavo Carneiro, Helder Fontes, and Manuel Ricardo. 2011. Fast prototyping of network protocols through ns-3 simulation model reuse. *Simulation Modelling Practice and Theory* 19, 9 (2011), 2063–2075. https://doi.org/10.1016/j.simpat.2011.06.002

[2] Pasquale Imputato and Stefano Avallone. 2016. Design and Implementation of the Traffic Control Module in Ns-3. In *Proceedings of the Workshop on Ns-3 (WNS3 '16)*. ACM, New York, NY, USA, 1–8. https://doi.org/10.1145/2915371.2915382

[3] Pasquale Imputato and Stefano Avallone. 2017. Traffic Differentiation and Multiqueue Networking in Ns-3. In *Proceedings of the Workshop on Ns-3 (WNS3 '17)*. ACM, New York, NY, USA, 79–86. https://doi.org/10.1145/3067665.3067677

[4] Luigi Rizzo. 2012. Netmap: A Novel Framework for Fast Packet I/O. In *Proceedings of the 2012 USENIX Conference on Annual Technical Conference (USENIX ATC'12)*. USENIX Association, Berkeley, CA, USA, 9–9. http://dl.acm.org/citation.cfm?id=2342821.2342830