

Validating Linux Network Emulation

Ezra Kissel

Department of Computer & Information Sciences
University of Delaware, Newark, DE 19716
kissel@cis.udel.edu

Martin Swany

School of Informatics and Computing
Indiana University, Bloomington, IN 47405
swany@iu.edu

1 Introduction

The commonly used Linux module *netem* [2] allows for the emulation of a wide range of network impairments, including latency, loss, packet re-ordering and duplication. While the observed behavior of *netem*-induced delay and loss effects on protocol performance has typically met our expectations, the absolute accuracy of the impairments have remained an open question. The evaluation described here makes use of a dedicated network impairment device, Spirent XGEM [1], to simulate a wide range of WAN conditions. This gives us a high degree of confidence that our results will emulate conditions on real-world networks, and provides a good basis with which to compare with the *netem* implementation.

2 Evaluation



Figure 1: 10G testbed with XGEM network emulator

Results were collected from a testbed environment consisting of 4 nodes separated by an XGEM device used to emulate real-world network impairment conditions, shown in Figure 1. The testbed formed a linear network topology with client and server nodes at the edges and netem nodes on each end of a network segment simulated by the XGEM. In this way, we could compare both *netem* and XGEM along the same path by either enabling or disabling impairments at each node. Each of the 4

nodes contained dual-port 10Gb/s Mellanox ConnectX-2 VPI adapters set to Ethernet link-layer mode. The netem systems were additionally outfitted with two Mellanox ConnectX-2 dual-port Ethernet NICs to provide connectivity across the XGEM. Each of the 4 systems were compute nodes with 8-core Intel Xeon CPUs and 189GB of RAM.

In this article, we are primarily interested in the accuracy of emulated latency and loss, thus we designed an experiment to compare both types of impairment individually and when operating together on the same node. The *netem* impairments were applied to ingress and egress interfaces on the netem nodes between the end hosts, while the XGEM was configured with the identical latency and loss configuration. Our test application was a simple client/server program running on our end hosts that started a single TCP connection and performed a memory-to-memory transfer for a duration of 60 seconds. We recorded the total number of bytes transferred and the average sustained rate for the duration of the connection. The results were averaged over a total of 20 runs. We also recorded the observed loss reported by an *iperf* [3] UDP test over the same path. The results are summarized in Table 1, including the mean and the coefficient of variation (CV) of bytes transferred between runs.

Overall, these results show that *netem* compares favorably with the XGEM over both latency and loss impairments. We do note that *netem* appears to be less consistent in its effect on protocol behavior, especially as significant loss is introduced. However, for latency behavior alone, *netem* is statistically comparable to the XGEM. For all but the most delicate experiments, we conclude that when correctly configured, *netem* provides a realistic estimation of impaired network conditions. These results

Test	Lat.	Loss	Mean (GB)	CV	Gb/s	Obs. Loss
No impair	0.6	0.0%	74.127	0.12%	9.88	0.0%
Netem	100	0.0%	71.293	0.22%	9.51	0.0%
Netem	0	0.01%	37.092	3.73%	4.95	0.0096%
Netem	100	0.01%	1.403	40.14%	0.187	0.0096%
XGEM	100	0.0%	71.192	0.17%	9.49	0.0%
XGEM	0	0.01%	35.233	1.73%	4.70	0.01%
XGEM	100	0.01%	1.378	17.41%	.184	0.01%

Table 1: Comparing network impairments applied with *netem* and XGEM.

provide confidence for its continued use as a less costly alternative to dedicated, hardware-based solutions.

References

- [1] Gem: Network impairment emulator. http://www.spirent.com/Solutions-Directory/Impairments_GEM.
- [2] Net:netem. <http://www.linux-foundation.org/en/Net:Netem>.
- [3] IPerf. <http://dast.nlanr.net/Projects/IPerf/>.