Chair of Network Architectures and Services
Department of Informatics
Technical University of Munich

Bachelor's Thesis: Performance-Analysis of VPP

Intermediate Talk

Presenter: Peter Okelmann— *Advisors:* Paul Emmerich, Dominik Scholz— *Supervisor:* Prof. Dr.-Ing. Georg Carle

VPP: a fast software router

VPP (Vector Packet Processing) is a user-space software router. It's approach combines many advantages:

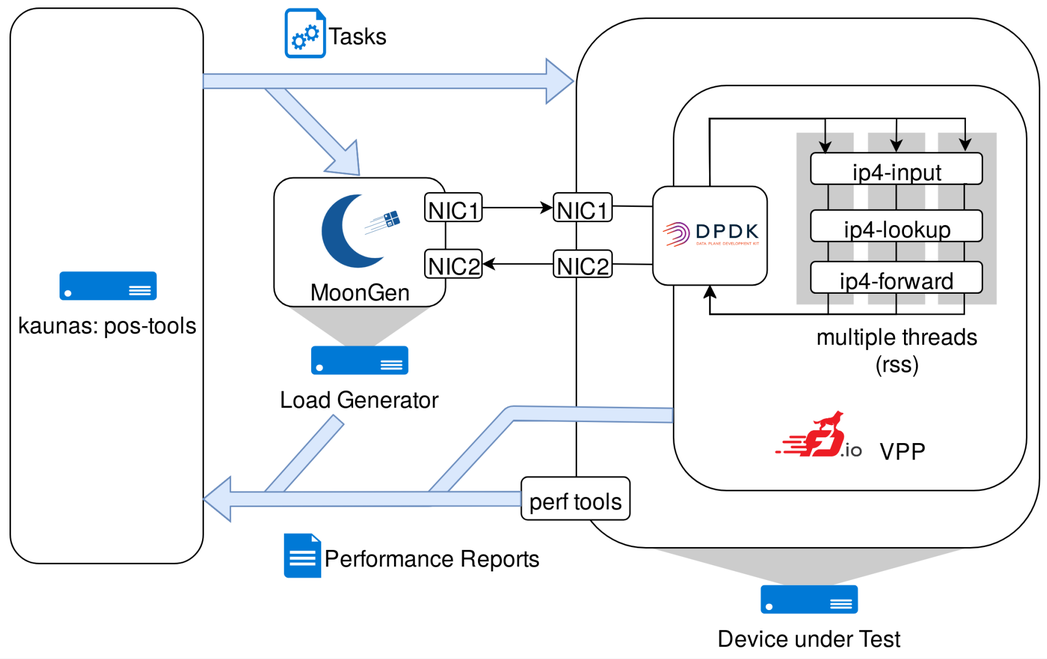
- deployable to mainstream architechtures
- fast, user-space NIC drivers
- can run in virtualized containers

"It is the open source version of Cisco's Vector Packet Processing (VPP) technology" [1] and is now beeing developed by FD.io ("The Fast Data Project") which belongs to the Linux Foundation.

Feature Highlights:

- vecorized processing of packets in badges
- utilizes high-speed dpdk drivers
- modular and extendable packet-processing graph [4]
- cpu-scalability

Benchmarking Setup



Testing Methodology

For tests to return meaningful latency results, the optimum of throughput to latency is beeing found by a script. This packet rate is then used for further tests. Otherwise a worst case of latency is triggered, because packet queues fill up. [3]

VPP properties to test:

- raw forwarding throughput
- latency: cache and memory impact
- packet processing graphs utilizing multiple cpu cores
- testing specific processing nodes / router features
- TODO

Testing Parameters

MoonGen [2] scripts can generate testing load according to the following testing parameters:

- packet rate
- packet size
- traffic type (generic Ethernet, UDP)
- traffic pattern (inter packet gaps)
- grant warmup time

Gathered testing results:

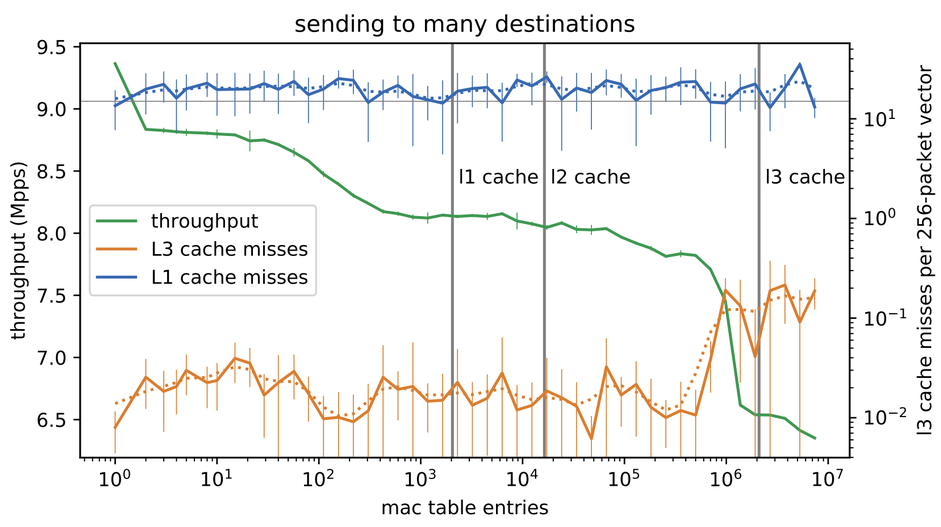
- latency histogram
- throughput
- linux perf stats (cache misses...)
- linux perf record (cpu-time spent per symbol)
- internal vpp state information

l2 Throughput

The following results are produced by sending non-ip ethernet packets to a single destination. Running VPP on an Intel E5-2640 @ 2.0GHz (cesis) with 10G networking results in the following numbers:

VPP config	max Mpps	stable Mpps	Relative
packet gen speed	14.86	14.86	100%
l2 xconnect	10.4	10.1	68%
l2 bridge: no features	9.35	9.2	62%
l2 bridge: mac-age	8.62	8.6	58%
l2 bridge: mac-learn	8.51	8.3	56%
l2 bridge: mac-learn, mac-age	8.50	8.3	56%

l2 Latencies per Flow



[1] What is vpp? https://wiki.fd.io/view/VPP/What_is_VPP%3F. Accessed on 2019-01-16.
[2] P. Emmerich, S. Gallenmüller, D. Raumer, F. Wohlfart, and G. Carle. MoonGen: A Scriptable High-Speed Packet Generator. In *Internet Measurement Conference 2015 (IMC'15)*, Tokyo, Japan, Oct. 2015.
[3] S. Gallenmüller, P. Emmerich, F. Wohlfart, D. Raumer, and G. Carle. Comparison of frameworks for high-performance packet io. In *Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for networking and communications systems*, pages 29–38. IEEE Computer Society, 2015.
[4] L. Linguaglossa, D. Rossi, S. Pontarelli, D. Barach, D. Marjon, and P. Pfister. High-speed software data plane via vectorized packet processing (extended version). *Tech. Rep.*, 2017.