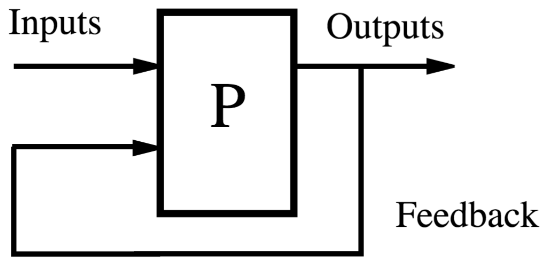


"**A feedback loop** where all outputs of a process are available as causal inputs to that process"

How to Push Extreme Limits of Performance and Scale with Vector Packet Processing Technology

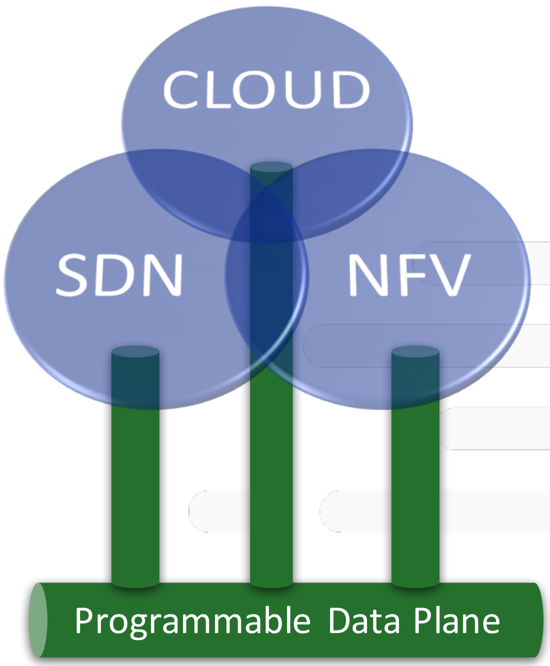
Maciek Konstantynowicz
FD.io CSIT Tech Project Lead

FD.io VPP Overview

Technology Benchmarking and Performance

Facilitating Test Driven Development

Evolution of Programmable Networking



- Many industries are transitioning to a more dynamic model to deliver network services
- The great unsolved problem is how to deliver network services in this more dynamic environment
- Inordinate attention has been focused on the non-local network control plane (controllers)
    - Necessary, but insufficient
- There is a giant gap in the capabilities that foster delivery of dynamic Data Plane Services

Copy from FD.io VPP materials: https://wiki.fd.io/view/Presentations

FD.io - Fast Data input/output – for Internet Packets and Services

What is it about – continuing the evolution of Computers and Networks:

- **Computers** => Networks => Networks of Computers => Internet of Computers
- **Networks** in Computers => Requires efficient packet processing in Computers
- **Enabling scalable modular Internet packet services in Computers** – routing, bridging and servicing packets
- Making Computers be part of the Network, making Computers become **a-bigger-helping-of-Internet**

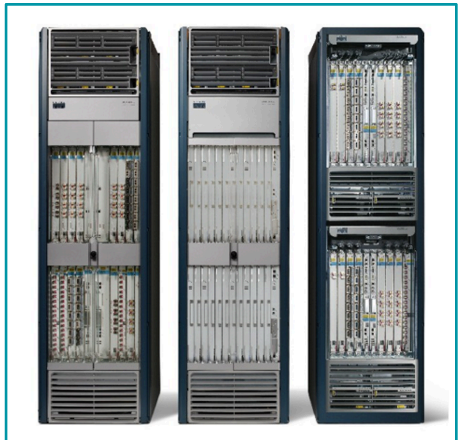
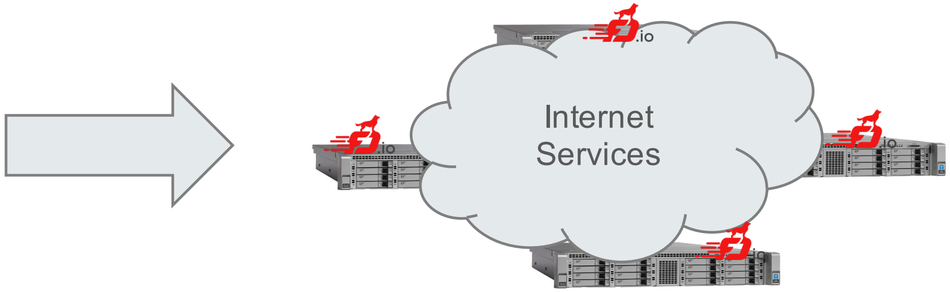


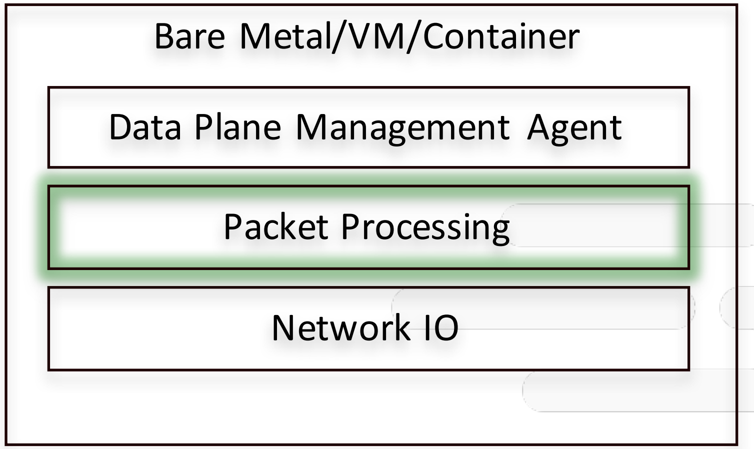
Figure 1. That's the image you get when you type "hfr" into your favorite Internet search engine, and search for images.



FD.io: www.fd.io

Blog: blogs.cisco.com/sp/a-bigger-helping-of-internet-please

Introducing Vector Packet Processor - VPP



- VPP is a rapid packet processing development platform for highly performing network applications.
- It runs on commodity CPUs and leverages DPDK
- It creates a vector of packet indices and processes them using a directed graph of nodes – resulting in a highly performant solution.
- Runs as a Linux user-space application
- Ships as part of both embedded & server products, in volume
- Active development since 2002

Copy from FD.io VPP materials: https://wiki.fd.io/view/Presentations

Aside: Benchmarking VNFs - sounds basic and straightforward. They are expected to behave like networking devices …

- **Start with the basics - Baseline the system**
  - Ensure repeatibility and consistency of results
  - Minimize uncertainties and errors
  - Understand and document the source of uncertaininties and errors
  - Quantify the amounts of uncertainty and errors

RFC 2330

- **Apply baseline network device(s) measurement practices**
  - Packet throughput across packet sizes
    - Focus on NDR (non drop rate)
  - Packet latency, latency variation

RFC 1242
RFC 2544
RFC 5481
draft-vsperf-bmwg-vswitch-opnfv

But …

Aside: Benchmarking VNFs - sounds basic and straightforward. They are expected to behave like networking devices ...

- Start with the basics - Baseline the system
  - Ensure repeatability and consistency of results
  - Minimize uncertainties and errors
  - Understand and document the source of uncertaininties and errors
  - Quantify the amounts of uncertainty and errors

RFC 2330

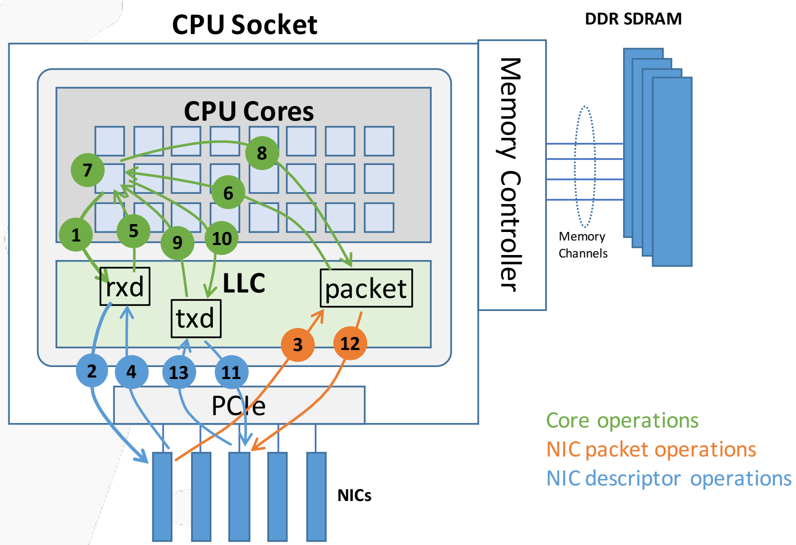
- Apply baseline network device(s) measurement practices
  - Packet throughput across packet sizes
    - Focus on NDR (non drop rate)
  - Packet latency, latency variation

RFC 1242
RFC 2544
RFC 5481
draft-vsperf-bmwg-vswitch-opnfv

But ... VNFs are not physical devices, they are SW workloads on commodity servers/CPUs

Network workloads vs. compute workloads

- **They are just a little BIT different – it's all about processing packets**
  - At 10GE, 64B frames can arrive at 14.88Mfps – that's 67nsec per frame.
  - With 2GHz CPU core clock cycle is 0.5nsec – that's 134 clock cycles per frame.
  - BUT it takes ~70nsec to access memory – too slow for required time budget.



- **Efficiency of dealing with packets within the computer is essential**
  - Moving packets: receiving on physical interfaces (NICs) and virtual interfaces (VNFs) => Need optimized drivers for both; should not rely on memory access.
  - Processing packets: Header manipulation, encaps/decaps, lookups, classifiers, counters => Need packet processing optimized for CPU platforms

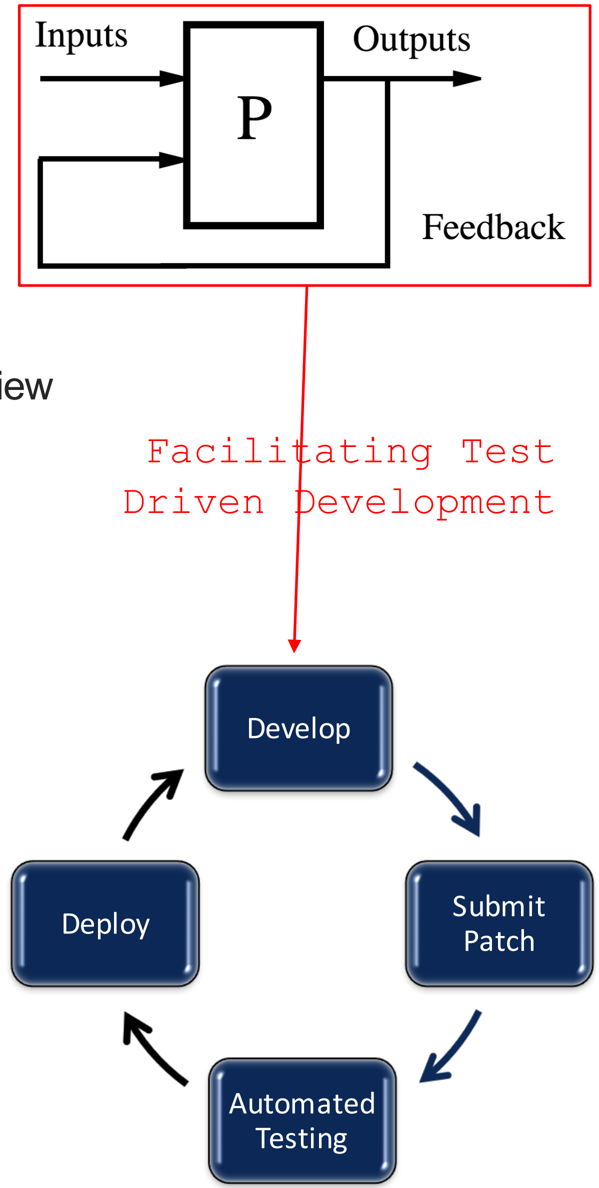
- **CONCLUSION - Must to pay attention to Computer efficiency for Network workloads**
  - Need to measure (count) instructions per packet for useful work (**IPP**)
  - Need to measure instructions per clock cycle (**IPC**)
  - Need to monitor cycles per packet (**CPP**)

Need reliable telemetry !!
(with representative and repeatable readings)

Not easy at Nx10GE, Nx40GE speeds, but possible..

FD.io Design Engineering by Benchmarking

Continuous System Integration and Testing (CSIT)



Fully automated testing infrastructure

- Covers both programmability and data planes
- Code breakage and performance degradations identified before patch review
- Review, commit and release resource protected

Continuous Functional Testing

- Virtual testbeds with network topologies
- Continuous verification of functional conformance
- Highly parallel test execution

Continuous Software and Hardware Benchmarking

- Server based hardware testbeds
- Continuous integration process with real hardware verification
  - Server models, CPU models, NIC models

More info: https://wiki.fd.io/view/CSIT

FD.io Continuous Performance Lab

a.k.a. The **CSIT** Project (**C**ontinuous **S**ystem **I**ntegration and **T**esting)

- **What it is all about – CSIT aspirations**
  - **FD.io VPP benchmarking**
    - VPP functionality per specifications (**RFCs**[1])
    - VPP performance and efficiency (**PPS**[2], **CPP**[3])
      - Network data plane - throughput Non-Drop Rate, bandwidth, PPS, packet delay
      - Network Control Plane, Management Plane Interactions (memory leaks!)
    - Performance baseline references for HW + SW stack (**PPS**[2], **CPP**[3])
    - Range of deterministic operation for HW + SW stack (**SLA**[4])
  - **Provide testing platform and tools to FD.io VPP dev and user community**
    - Automated functional and performance tests
    - Automated telemetry feedback with **conformance**, **performance** and **efficiency** metrics
  - **Help to drive good practice and engineering discipline into FD.io VPP dev community**
    - Drive innovative optimizations into the source code – verify they work
    - Enable innovative functional, performance and efficiency additions & extensions
    - Make progress faster
    - Prevent unnecessary code "harm"

Legend:

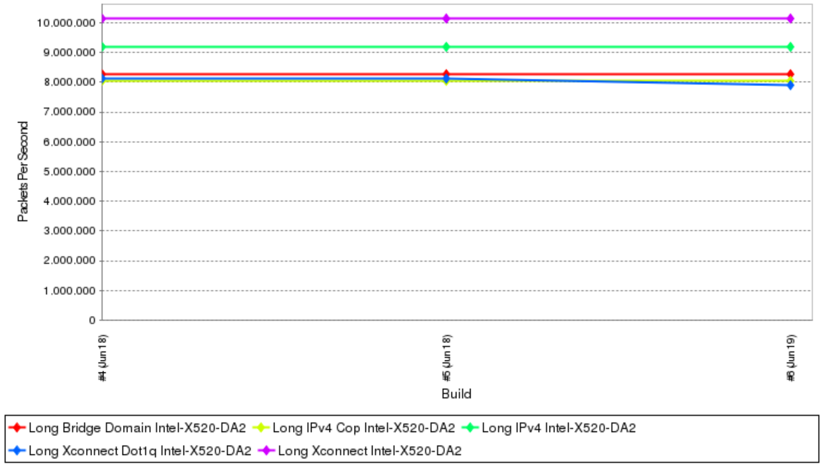
[1] RFC – Request For Comments – IETF Specs basically
[2] PPS – Packets Per Second
[3] CPP – Cycles Per Packet (metric of packet processing efficiency)
[4] SLA – Service Level Agreement

FD.io

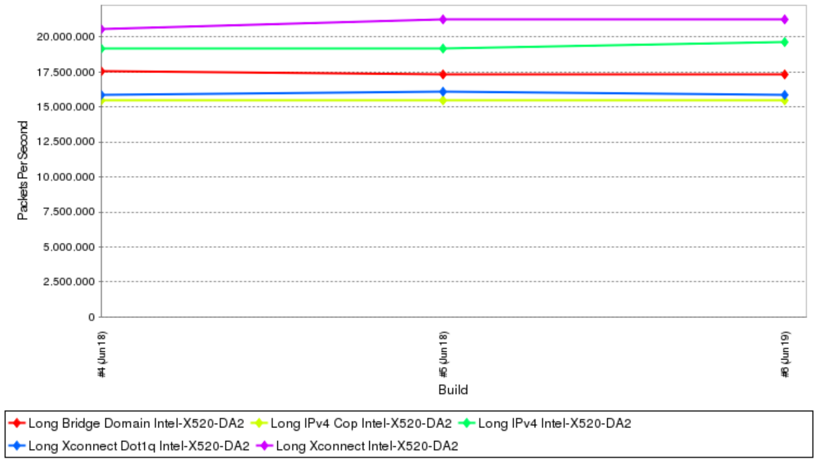
CSIT/VPP-v16.06 Report

- **Testing coverage summary**
  - L2, IPv4, IPv6
  - Tunneling
  - Stateless security
- **Non Drop Rate Throughput**
  - 8Mpps to10Mpps per CPU core at 2.3GHz*
  - No HyperThreading
- **Improvements since v16.06**
  - 10Mpps to 12Mpps per CPU core at 2.3GHz*
  - With HyperThreading gain ~10%

RFC2544 binary search with 64B, worker-thread=1, rss=1, NDR



RFC2544 binary search with 64B, worker-thread=2, rss=1, NDR



https://wiki.fd.io/view/CSIT/VPP-16.06_Test_Report

*CPU core 2.3GHz – Intel XEON E5-2699v3, https://wiki.fd.io/view/CSIT/CSIT_LF_testbed

Measurement problems encountered …

The learning curve

Not basic and straightforward at all

Need to apply knowledge of the overall system – know your complete Hardware and Software stack (cross-disciplinary).

- **Problem:**
  - Throughput - test trials yielding non-repeatable results, including RFC2544 tests.
- **Resolution - identify and quantify system-under-test bottlenecks**
  - HW: NIC, PCI lanes, CPU sockets, Memory channels.
    - Operate within their deterministically working limits - make sure they are not DUTs :)
    - Intelligent CPUs – control their "intelligence" !
  - OS: kernel modules interferring with tests by using shared resources e.g. CPU cores
    - Isolate CPUs, avoid putting DUT workloads on non-isolated cores.
    - Still kernel is interferring - more on this later.
  - VM environment:
    - Hypervisor entries/exits: hard to track the impact, but not impossible, just labour intensive - combinatorial explosion of things to test doesn't help !
- **Adjust testing methodologies**
  - RFC2544 binary search start/stop criteria – LowRate-to-HighRate, HighRate-to-LowRate.
  - Linear throughput, packet loss scans.

Measurement problems encountered …

The learning curve

Not basic and straightforward at all

Need to apply knowledge of the overall system – know your complete Hardware and Software stack (cross-disciplinary).

- **Problem:**
  - Packet latency and latency variation vary greatly across tested VNF systems.
  - Min/max/avg latency and latency variation (jitter) measurements not enough; they hide periodic latency spikes, and packet latency patterns.
  - Lack of tools to measure and report per packet latency under throughput load.
- **Resolution (work in progress)**
  - In discussion with HW tester vendors, but progress slow.
  - Exploring options for developing own Software based tools to address the gap
    - Doing it at Nx10GE, nx40GE is challenging but feasible ☺

Measurement tools …
Need more, need better

- **Problem:**
  - HW testers expensive, not flexible, not easy to integrate into CI/CD systems

- **Resolution (work in progress)**
  - Use Software based packet generators and testers
  - Challenges:
    - Accurate latency measurements
    - PPS and Gbps scale - doing it at Nx10GE, nx40GE is challenging but feasible ☺

Computer HW telemetry tools …

Need more, need better

- **Problem:**
  - Modern computers/CPUs provide lots of telemetry data and performance counters
  - Challenge – readings not always repeatable, which ones do you trust

- **Resolution (work in progress)**
  - Work with CPU hardware vendors to interpret the counters
  - Drive development of open-source SW tools for computer/CPU performance monitoring and reporting
  - It can only get better ☺

To Dos

- Address per packet latency and latency variation measurements
- Automate detection of packet throughput and latency inconsistencies
- Work with community and vendors on improving network-centric telemetry tools for computers/CPUs
  - Counters accuracy
  - Reporting clarity
  - Measurements repeatibility
- Work with IETF ippm and bmwg on standardizing best practices of automated vNF benchmarking
  - Describing the tests using data model language (YANG) is really really cool!
  - Key for driving standardized test automation

Q&A