CSIT TOI

CSIT Project Overview

Maciek Konstantynowicz

Feb 6, 2018

CSIT TOI Schedule

- **TOI-1 06-Feb-2018**
  - CSIT overview, goals, releases, high-level design - 10min Maciek K.
  - CSIT libraries, L1/L2 KWs, sample test suites - 20min Tibor F.
  - Performance tests - 20min – Peter M.
  - Q&A - 10min
- **TOI-2 08-Feb-2018**
  - VIRL functional tests - 20min – Jan G.
  - Physical testbeds - 10min – Peter M.
  - VIRL testbeds - 10min – Ed K.
  - Q&A - 20min

CSIT Overview - Topics

- Project scope
- CSIT high-level design
- Test coverage
- Where we want to take it..
- Joining CSIT project
- References

FD.io Continuous Performance Lab

a.k.a. The CSIT Project (Continuous System Integration and Testing)

- **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 usr 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 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

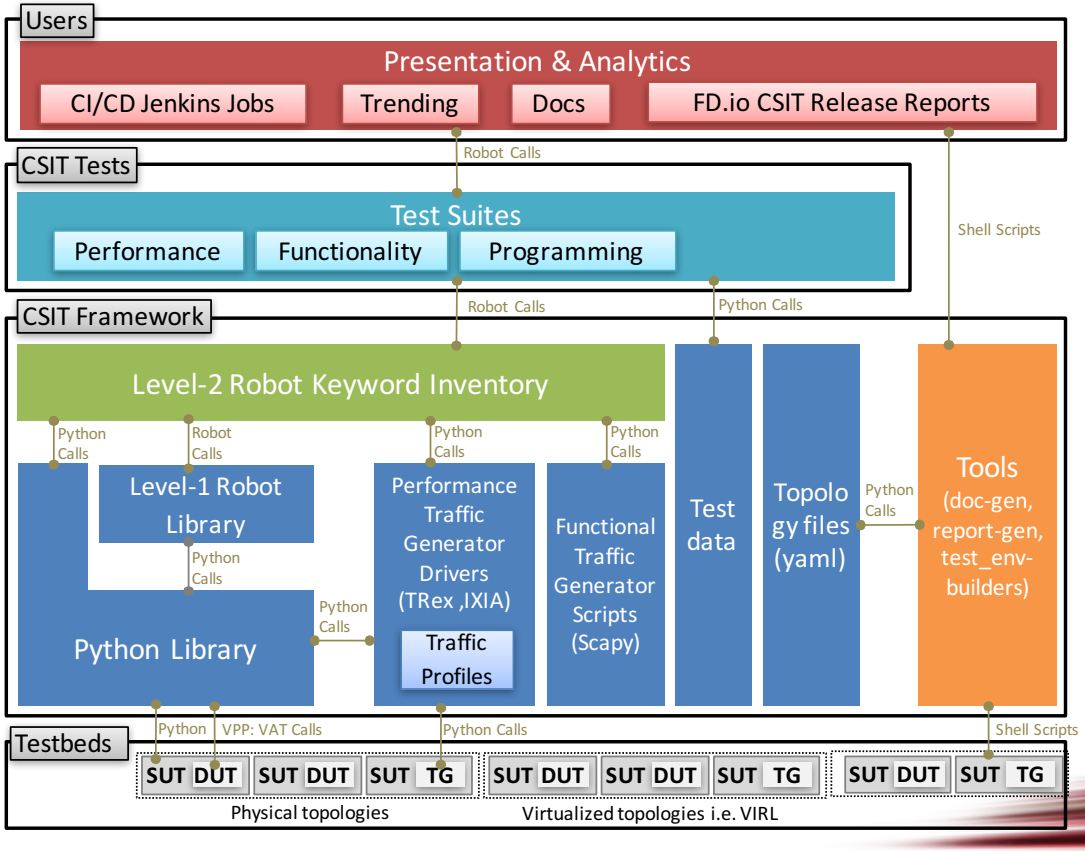
CSIT Project Scope

1. **Fully automated testing of LF[1] FD.io[2] systems**
   a. FD.io VPP[3] and related sub-systems (e.g. DPDK[5] Testpmd[6], Honeycomb[7], ...)
   b. Functionality, performance, regression and new functions.
2. **Functionality tests - against functional specifications**
   a. VPP data plane, network control plane, management plane.
3. **Performance tests - benchmarking data plane and control plane**
   a. VPP data plane incl. non-drop-rate and partial-drop-rate packet throughput and latency.
   b. Network control plane, management plane.
4. **Test definitions driven by FD.io VPP functionality, interfaces and performance**
   a. Uni-dimensional tests: data plane, network control plane, management plane.
   b. Multi-dimensional tests: Use case driven.
5. **Integration with LF VPP test execution environment**
   a. Performance tests execute in physical compute environment hosted by LF.
   b. Functional tests execute in virtualized VM VIRL[9] environment hosted by LF.
6. **Integration with LF FD.io CI[8] system including FD.io Gerrit and Jenkins**
   a. Test auto-execution triggered by VPP verify jobs, periodic CSIT jobs, and other FD.io jobs.

(1) LF - Linux Foundation.
(2) FD.io - Fast Data I/O - project in LF.
(3) VPP - Vector Packet Processing - sub-project in FD.io.
(5) DPDK - Data Plane Development Kit.
(6) Testpmd - DPDK example application for baseline DPDK testing.
(7) Honeycomb - Model-driven management agent for VPP - project in FD.io.
(8) CI - Continuous Integration.
(9) VIRL - Virtual Internet Routing Lab, SW platform donated by Cisco.

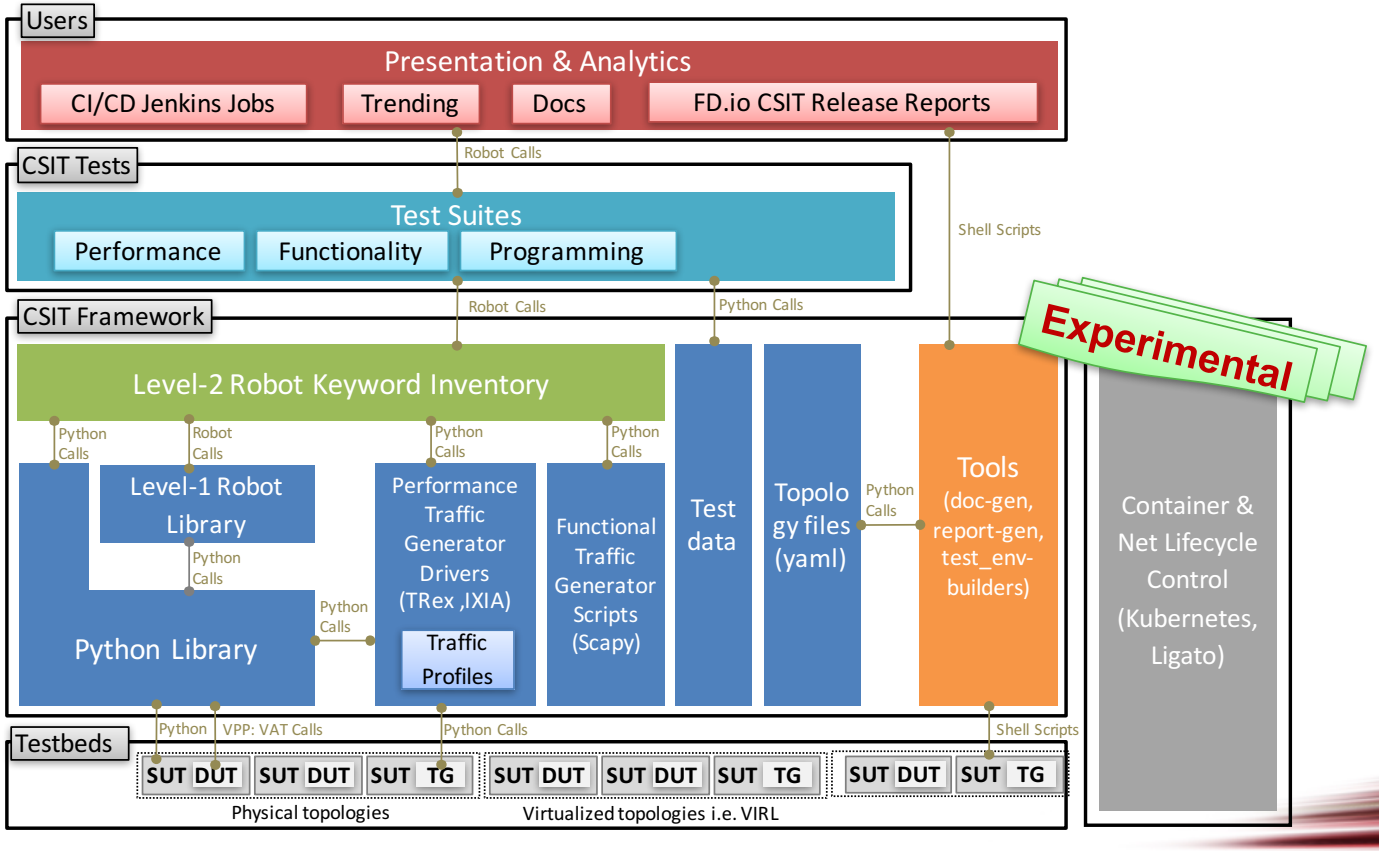
CSIT high-level design

- Layered CSIT system design
  - Users, tests, framework, testbeds
  - Abstracts test topologies to support variety of environments
- High degree of design modularity
  - Easy to expand and plug new functionality
- Large number of performance and functional tests
  - Functional in virtualized topologies: VPP, HoneyComb, NSH_SFC
  - Performance in physical topologies: VPP, DPDK-testpmd, VPP with VMs, VPP with Containers
- Automated test results analytics and report generation
  - Goal: Faster detection of anomalies and bad trends



CSIT high-level design – recent additions

- Orchestrated container networking topologies
  - Based on Kubernetes and Ligato projects
  - Virtual topologies with VPP memif: “parallel”, “chain”, “horizontal”
  - Experimental
- VPP TCP/IP tests
  - With **wrk** HTTP benchmarking tool
- ARM micro-architecture support



VPP Functional Test Coverage*

- DHCP - Client and Proxy
- GRE Overlay Tunnels
- L2BD Ethernet Switching
- L2XC Ethernet Switching
- LISP Overlay Tunnels
- Softwire Tunnels
- Cop Address Security
- IPSec - Tunnels and Transport
- IPv6 Routed-Forwarding
- uRPF Source Security
- Tap Interface
- Telemetry - IPFIX and SPAN
- VRF Routed-Forwarding
- iACL Security
- IPv4 Routed-Forwarding
- QoS Policer Metering
- VLAN Tag Translation
- VXLAN Overlay Tunnels

* https://docs.fd.io/csit/rls1710/report/vpp_functional_tests/overview.html#functional-tests-coverage

VPP Performance Test Coverage*

- L2 switching: cross-connect, bridge-domain incl. MAC learning @scale
- IPv4, IPv6 routed-forwarding: base and scale /32 /128
- Overlays: lisp, gpe, vxlan
- Acls: ip4, stateless, stateful
- Nat: cgnat44, nat44, based and scale
- Crypto: QAT HW, SW
- VM Vhost topologies
- LXC and K8s orchestrated container memif topologies

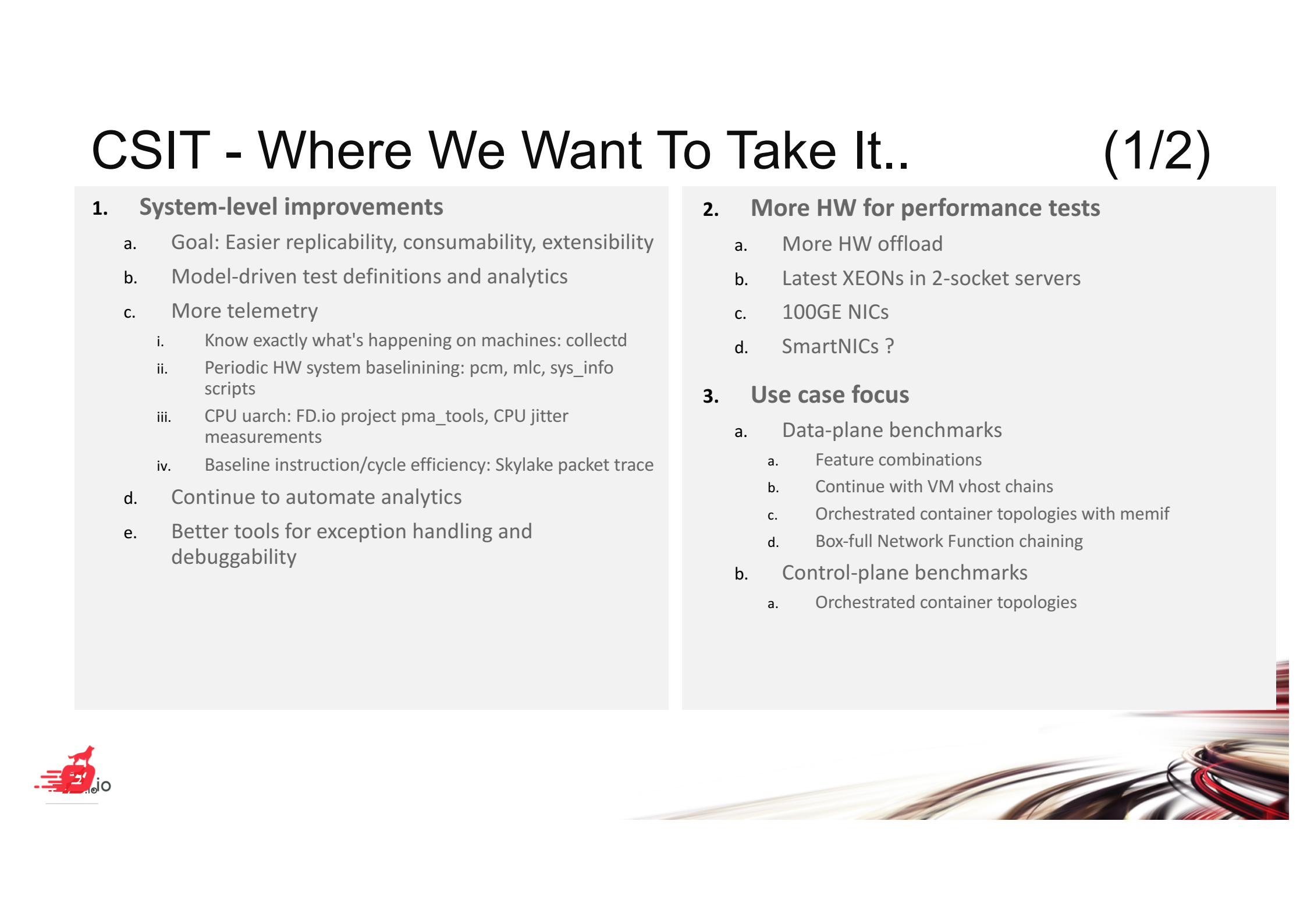
* https://docs.fd.io/csit/rls1710/report/vpp_performance_tests/overview.html#performance-tests-coverage

CSIT - Where We Want To Take It.. (1/2)

1. **System-level improvements**
   a. Goal: Easier replicability, consumability, extensibility
   b. Model-driven test definitions and analytics
   c. More telemetry
      i. Know exactly what's happening on machines: collectd
      ii. Periodic HW system baselinining: pcm, mlc, sys_info scripts
      iii. CPU uarch: FD.io project pma_tools, CPU jitter measurements
      iv. Baseline instruction/cycle efficiency: Skylake packet trace
   d. Continue to automate analytics
   e. Better tools for exception handling and debuggability

2. **More HW for performance tests**
   a. More HW offload
   b. Latest XEONs in 2-socket servers
   c. 100GE NICs
   d. SmartNICs ?

3. **Use case focus**
   a. Data-plane benchmarks
      a. Feature combinations
      b. Continue with VM vhost chains
      c. Orchestrated container topologies with memif
      d. Box-full Network Function chaining
   b. Control-plane benchmarks
      a. Orchestrated container topologies



CSIT - Where We Want To Take It.. (2/2)

4. **Providing feedback to directly associated projects**
   a. FD.io VPP
   b. FD.io Honeycomb
   c. DPDK Testpmd
5. **Making Test-verified Use Recommendations for FD.io**
6. **Providing feedback to technology stack owners:**
   a. HW CPU: CPU technologies e.g. Intel
   b. HW NICs: HW offload, driver costs, ...
   c. OS: kernel related e.g. CFS
   d. Virtualization: QEMU, libvirt, ...
   e. Containerization: LXC, Docker, …
   f. Orchestration integration: Kubernetes, ...
7. **Asking technology stack owners for optimizations**

Joining CSIT Project and Contributing..

- Visit FD.io CSIT wiki: https://wiki.fd.io/view/CSIT

Get Involved [edit]
- Weekly CSIT Meeting.
- Join the CSIT Mailing List.
- Join fdio-csit IRC channel.

Start Here [edit]
- What is CSIT (Continuous System Integration and Testing)?
- CSIT System Design
- Getting started with CSIT development
- Submitting Patches to CSIT
- CSIT Test Naming
- CSIT Development Tasks (Jira)
- CSIT Jenkins Jobs
- CSIT Branching Strategy
- CSIT VIRL Testbed
- CSIT Performance Testbed

CSIT Code Documentation [edit]
- CSIT master Documentation
- CSIT 17.10 Documentation
- CSIT 17.07 Documentation
- CSIT 17.04 Documentation
- CSIT 17.01 Documentation
- CSIT 16.09/16.06 Documentation

Test Development Plans [edit]
- CSIT 18.01 Plan

Test Reports [edit]
- CSIT rls 1710 Test Report
- CSIT rls 1707 Test Report
- CSIT rls 1704 Test Report
- CSIT rls 1701 Test Report
- CSIT/VPP-16.09_Test_Report
- CSIT/VPP-16.06_Test_Report

Join and contribute.

Get familiar with CSIT design, operations and testing environments .

Get familiar with code documentation and reports.
Fdio CSIT releases (code and reports) follow VPP release cycle.

References

CSIT and related FD.io projects

- CSIT - Continuous System Integration and Testing: https://wiki.fd.io/view/CSIT
- VPP - https://wiki.fd.io/view/VPP
- pma_tools - https://wiki.fd.io/view/Pma_tools

Benchmarking Methodology

- “Benchmarking and Analysis of Software Network Data Planes” by M. Konstantynowicz, P. Lu, S.M. Shah, https://fd.io/resources/performance_analysis_sw_data_planes.pdf

Benchmarks

- EEMBC CoreMark® - http://www.eembc.org/index.php
- DPDK testpmd - http://dpdk.org/doc/guides/testpmd_app_ug/index.html
- FDio VPP – Fast Data IO packet processing platform, docs: https://wiki.fd.io/view/VPP, code: https://git.fd.io/vpp/

Performance Analysis Tools

- “Intel Optimization Manual” – Intel® 64 and IA-32 architectures optimization reference manual
- Linux PMU-tools, https://github.com/andikleen/pmu-tools

TMAM

- Intel Developer Zone, Tuning Applications Using a Top-down Microarchitecture Analysis Method, https://software.intel.com/en-us/top-down-microarchitecture-analysis-method-win
- Technion presentation on TMAM , Software Optimizations Become Simple with Top-Down Analysis Methodology (TMAM) on Intel® Microarchitecture Code Name Skylake, Ahmad Yasin. Intel Developer Forum, IDF 2015. [Recording]
- A Top-Down Method for Performance Analysis and Counters Architecture, Ahmad Yasin. In IEEE International Symposium on Performance Analysis of Systems and Software, ISPASS 2014, https://sites.google.com/site/analysismethods/yasin-pubs

THANK YOU !