InBand Network Telemetry

With P4 and FPGA at 100 Gbps

Viktor Puš, CESNET (pus@cesnet.cz)

2017-06-07, DXDD, Utrecht

Why INT?

Classical network monitoring: NetFlow/IPFIX

- Collect statistics about packets and bytes per network connection
  - Connection = set of packets sharing the same 5-tuple (SRC IP, DST IP, Proto, SRC Port, DST Port)
- Collected and exported by switches, routers, or dedicated boxes
- We get valuable data about L3 and L4 (sometimes L7)
- But we know very little about the underlying L2 infrastructure!
  - Overloaded lines, packet drops, latencies
- Switches hold this information - how can we retrieve it?
  - SNMP is 1988, slow, insecure, lacks detail

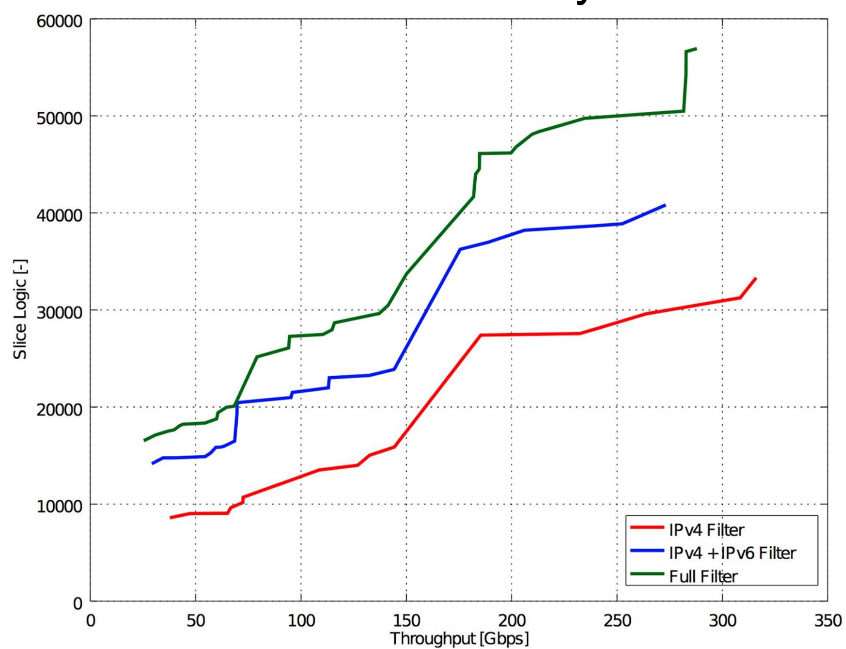
Inband Network Telemetry!

What is INT?

- "Inband Network Telemetry is a framework designed to allow the collection and reporting of network state by the data plane, without requiring intervention or work by the control plane."[1]
- Packets carry dedicated INT headers, added by switches
  - Detailed info about each packet's journey
    - Path, per-switch latency, queue occupation, ...
  - New protocol, not standardized
  - Need switch and endpoint support
  - Perfect use case for P4
- Our goal: Ultimate INT endpoint and analytics

CESNET

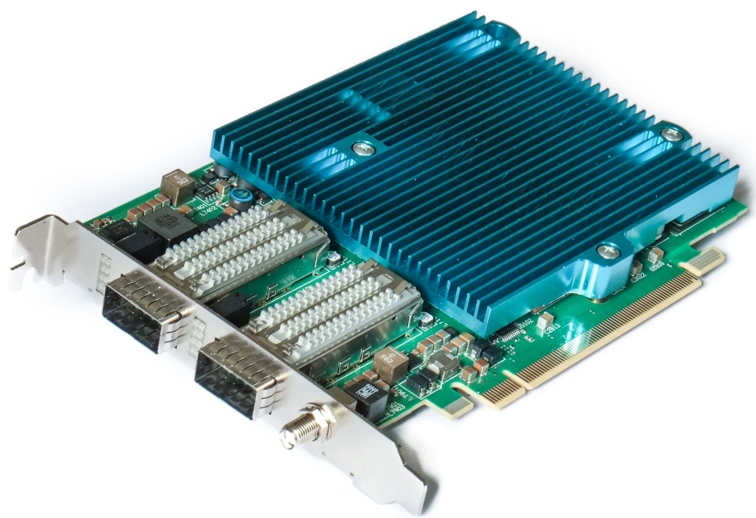
- Czech NREN, 100G network, ~400k users
- Liberouter research team
  - Network acceleration since 2003
  - Applications: Monitoring, Security, DDoS Mitigation
  - Technologies: FPGA, 100GE, PCI Express, DPDK
  - **Compiler from P4 to VHDL**
    - Hardware acceleration made easy for network/security experts



Commercial Partners

NETCOPE TECHNOLOGIES

- Products for the hardware acceleration of network traffic processing using FPGA
- Contribution: NFB-100G2

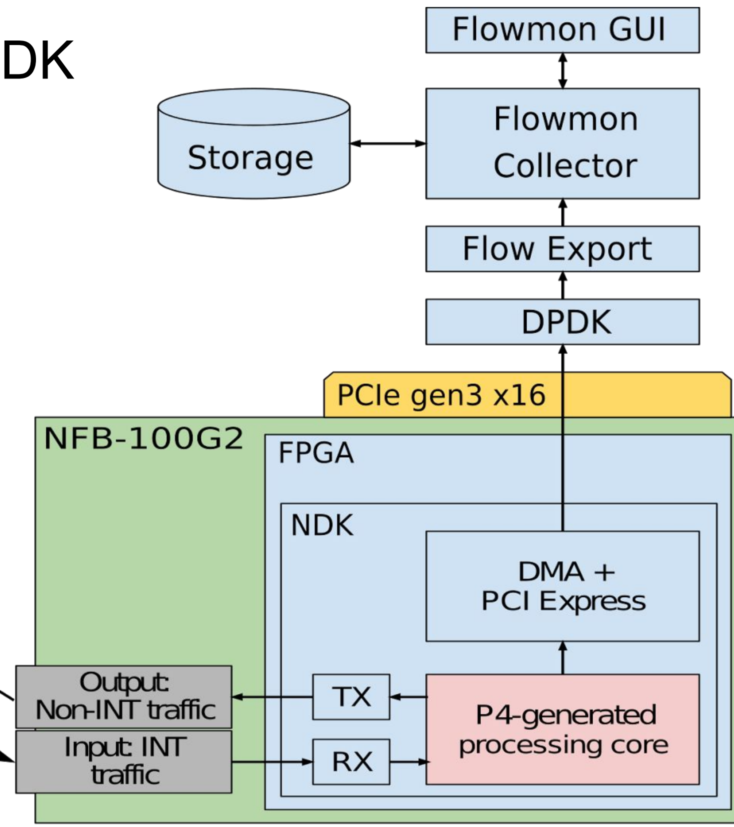


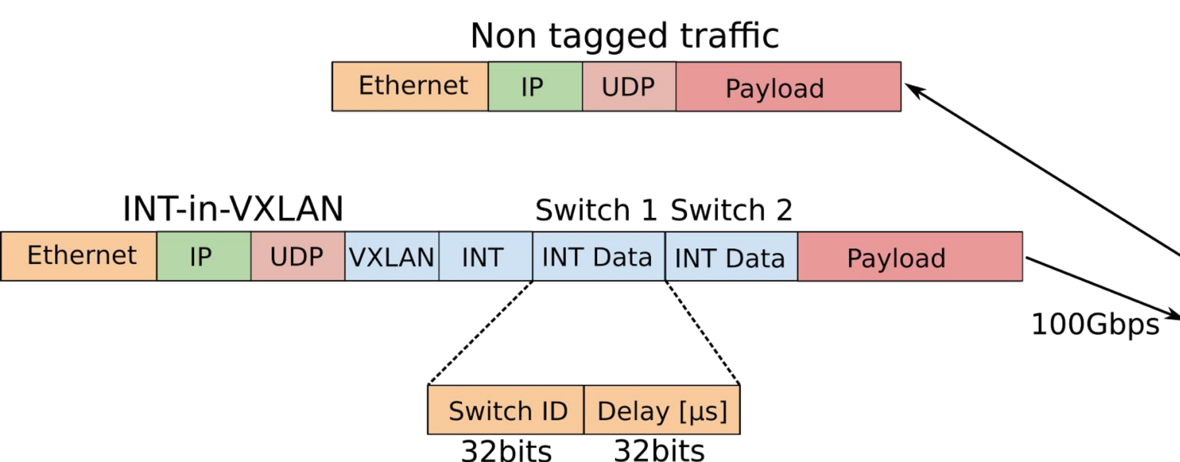
Flowmon Networks

- Everything needed to get complete network traffic visibility and analysis
- Contribution: Flow Exporter and Collector SW

INT@100Gbps

- INT-in-VXLAN traffic is pre-generated in PCAPs and sent via 2nd FPGA card at 100 Gbps
- FPGA receives INT-tagged traffic
- Removes INT headers and sends “original” packets out at line rate
  - Destination is not aware of INT
- Relevant fields are forwarded to SW via DPDK
  - P4 `generate_digest` action
- Flow Exporter generates NetFlow records and sends them to Collector
- Visualization and query interface





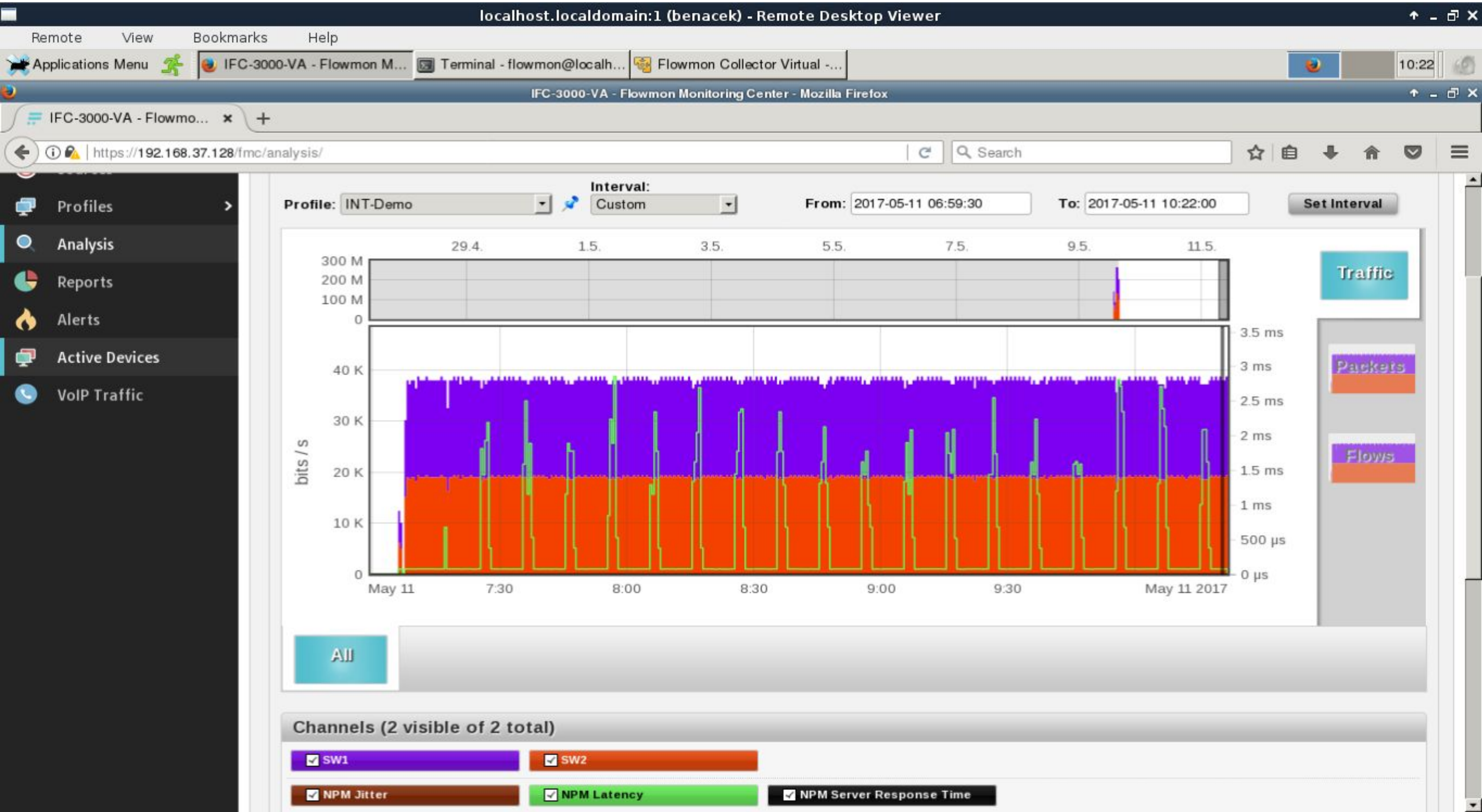
Console interface

```
RX Statistics:

                                         |          RX0          |
------------------------------------------------------------------
Packets [-]                              |      65097920011      |
Discarded [-]                            |           0           |
Octets [B]                               |    47653324455398     |
Throughput [Gbps]                        |    96.4081850653      |


TX Statistics:

                                         |          TX0          |
------------------------------------------------------------------
Packets [-]                              |      65099452205      |
Octets [B]                               |    42837086559877     |
Throughput [Gbps]                        |    86.6098969874      |


Duration: 1 h: 6 m: 13 s
```

```
         Switch 0: min_delay=40 us, max_delay=52 us, average_delay = 44.67
         Switch 1: min_delay=80 us, max_delay=114 us, average_delay = 93.33

Thu May 11 10:20:30 2017

         Switch 0: min_delay=49 us, max_delay=55 us, average_delay = 51.67
         Switch 1: min_delay=96 us, max_delay=112 us, average_delay = 105.33

Thu May 11 10:20:32 2017

         Switch 0: min_delay=47 us, max_delay=56 us, average_delay = 52.75
         Switch 1: min_delay=106 us, max_delay=116 us, average_delay = 110.50
```

Flowmon Web GUI

Graphs, queries



Conclusion

- P4 shortens development time
  - **New application**, running at **100 Gbps** and integrated into **commercial-grade** solution in about **3 weeks**
    - PCAP data (Python-scapy)
    - Firmware core (P4)
    - Flowmon Exporter input plugin (C)
    - Flowmon Collector modification (C+PHP)
  - No expert hardware knowledge needed
- Synthesis from P4 doesn’t have negative impact on unique FPGA features
  - High (and guaranteed) throughput
    - Bandwidth = bus width × frequency
  - Constant latency
  - Easy extensions for unanticipated functions

Thank you!

Web: www.liberouter.org

Twitter: @liberouter

P4 Workshop 2017-05