PLUS-aware Middlebox Implementation with FD.io/VPP

Tobias Bühler, ETH

MAMI Plenary Oslo, 4-5 July 2017

measurement and architecture for a middleboxed internet

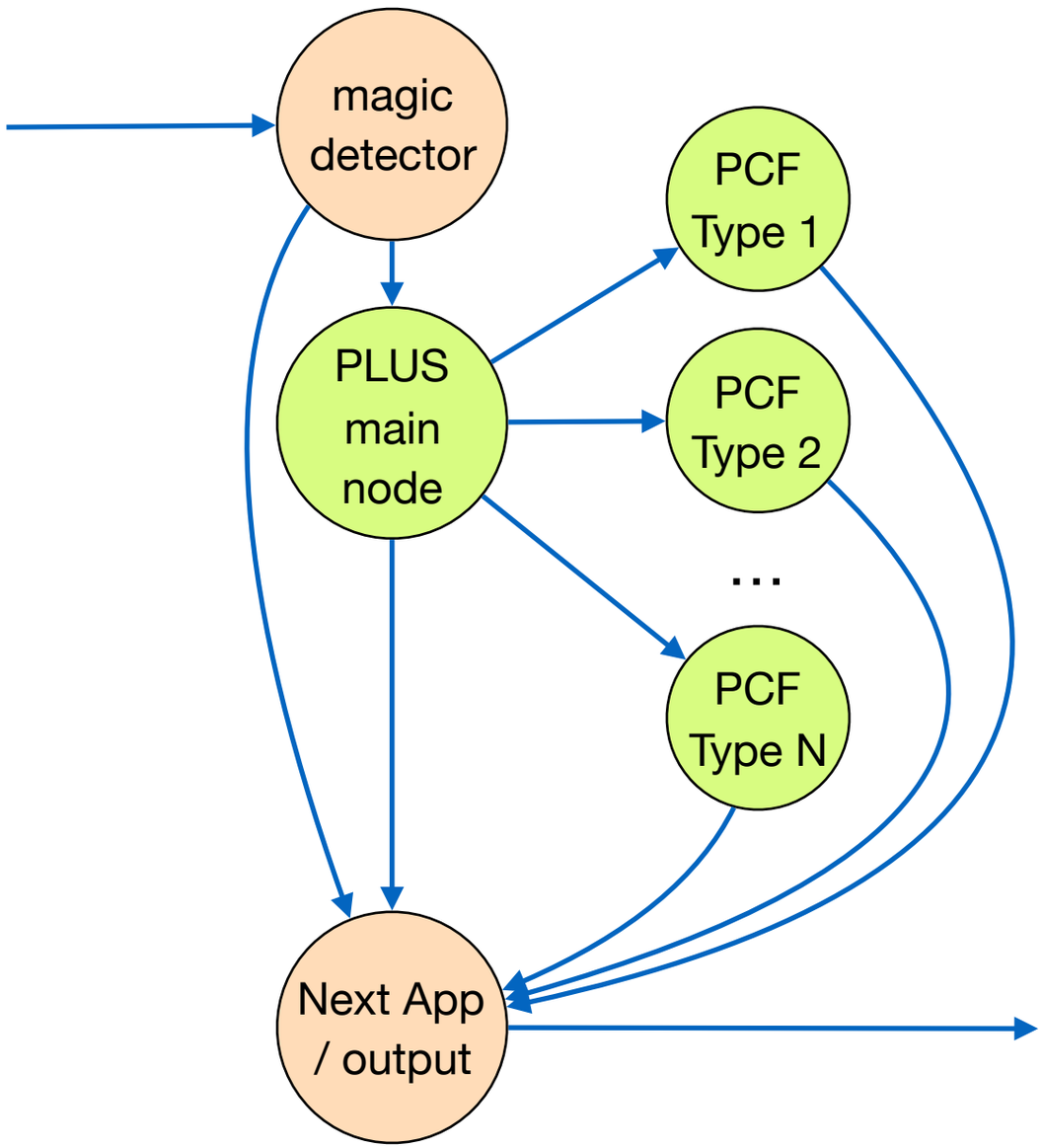
measurement | architecture | experimentation

FD.io / VPP

- FD.io is mainly based on Cisco's Vector Packet Processing (VPP) library
  - Open source implementation, highly optimized C code
  - Each VPP node has a packet buffer
    - Easy to modify a packet header field (if the size does not change)
    - If possible, two packets are processed at the same time
- Additionally, each packet has corresponding metadata
- FD.io can be deployed on real hardware or in a VM (Vagrant image available)

PLUS implementation

- 3 main steps:
  - Detect PLUS magic number
  - PLUS main node
  - PLUS extended header node per PCF type



PLUS detection

- Requires small code changes in an existing VPP node
  - Multiple candidates available (e.g. UDP lookup node)
- Check if the current packet has at least 4 bytes available after the UDP header
- Fetch the 4 bytes following the UDP header
- Compare these bytes with the PLUS magic number
- If a PLUS header is available: forward packet to the **PLUS main node**

PLUS main node

- Handles state update and timers for the current packet
  - FD.io provides various bihash functions
  - CAT & 5-tuple can be saved in 24 bytes (IPv4) or 48 bytes (IPv6)
- Saves total PLUS header size in packet metadata
  - Important for following nodes (may not be aware of PLUS header)
- Looks for an extended PLUS header
  - Packet is forwarded to corresponding VPP node based on the PCF Type

Extended header node

- Each PCF Type has its own VPP node
  - New extended header types can be added easily
- Current sample implementation: **PLUS-aware hop count**
  - PCF Type 0x01, 8 bit hop counter
  - Integrity Indicator (II) is 00 (no integrity protection)
  - Initially, the PCF Value is zero
  - Each PLUS-aware middlebox increases the counter by one

General FD.io findings

- It is quite easy to add your own VPP node
  - Each VPP node has a similar structure (skeletons available)
  - A lot of existing libraries for state, timers, and so on
  - Included „for free“:
    - Packet tracing through the entire VPP tree
      - Each node contributes a customizable status message
    - Detailed performance metrics
      - Per-node packet counters and CPU usage

Next steps

- Support for additional extended header types
- LoLa and RoI support and measurements
- Integration with load-balancing
- Detailed performance measurements on real hardware