INTRODUCTION

Today’s network programmability mainly provided by OpenFlow:

- Controller is logically-centralized
- Abstraction over a fixed set of dataplane functions (packet matching, counters)
- Simplifies Network Management, Evolution and Innovation

- Lots of work in Routing, Traffic Engineering, Quality of Service enforcement (QoS) or Network Virtualization have leveraged OpenFlow.

OpenFlow is the de facto Implementation of Software Defined Networking but only marginally covers the SDN concept:

- Programmability is very limited to a fixed set of features
- supported protocols, statistics, actions ...
- Dataplane still remains a blackbox

- Upcoming and custom network protocols (SCTP, QUIC, VxLAN ...)
- Custom Routing, Queuing, Load-Balancing, QoS enforcement
- Custom Statistics and Notifications for telemetry

APPRAOCH

Platform and Protocol-Independent Instruction set:

- The [e]BPF instruction set has been widely used for packet processing, filtering, and classification
  - Linux Kernel, FreeBSD, TCPdump/LibPCAP, Wireshark ...
  - Platform-independent RISC-like instruction set designed specifically to be protocol independent and used for real-time packet processing with fixed time constraints by disabling loops.
  - Language support for hashtable lookup, adding support for TCAM an LPM table lookup is under discussion.

Move away from the typical match-action pipeline:

- Replace linear, multi-stage match-action pipeline with many instances of eBPF execution units; high programmability at high aggregate throughput
- Use a higher-level programming language to express dataplane behaviour
  - P4 to eBPF compiler
  - Restricted C to eBPF compiler

REFERENCES