M-CORD
Vendor Independent Acceleration as a Service (XaaS)

Nic Viljoen

CORD
Central Office Re-architected as a Datacenter

#OpenCORD

The Problem

- The drive for vendor independence is causing a shift to the use of COTS equipment within the carrier data center
  - This is the driver that has led to the disaggregation of software and hardware solutions

- However due to some disruption in Moore's Law (as stated by Intel) the CPU is not scaling to the next generation of workloads
  - Jon Donovan of AT&T estimated traffic passing through their network has increased by 150,000% in the last 9 years
  - CPU has not scaled at the same rate, as Intel recently said within the MIT technology review, Moore's Law is breaking down

- This means that acceleration agents are required
  - However, the previous generation of ASIC-based acceleration does not fit the software-defined model
  - This is why networking is now being driven to a similar co-processing model such as graphics

- This however means that the issue of vendor independence returns
  - *If vendors are supplying proprietary software to enable the use of their own server accelerators then we are back to proprietary systems*
  - *This is the key problem that accelerators need to solve*

- How can we create a system whereby acceleration hardware and software are disaggregated?

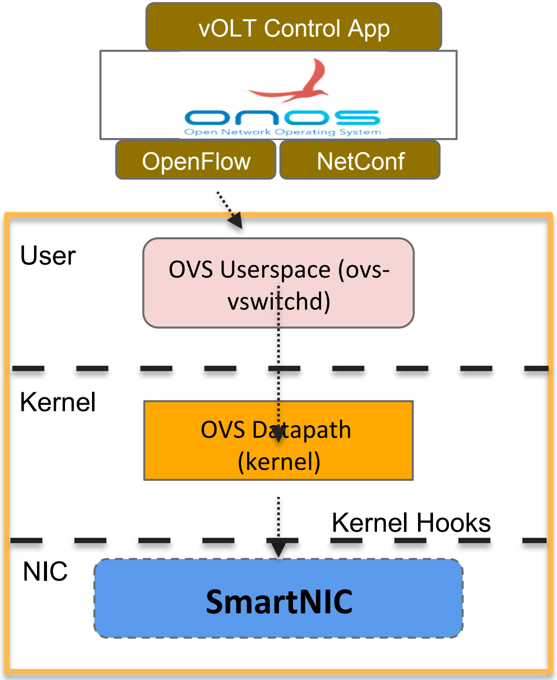
What is required for vendor independence?

- The acceleration device should be easily replaceable
  - If an accelerator is placed directly on the motherboard then once again the server is no longer COTS and vendor lock in is once again present
  - Implement as server-based NIC
- The software written should work (relatively) seamlessly across different accelerators
  - Software should be fully portable, to ensure plug and play functionality is not impaired
- The software written should perform without any added hardware acceleration
  - This requires a model which allows for acceleration both in CPU or with hardware acceleration

vSwitch Acceleration

➢ vSwitch acceleration is an early production ready case of attempting transparent offload

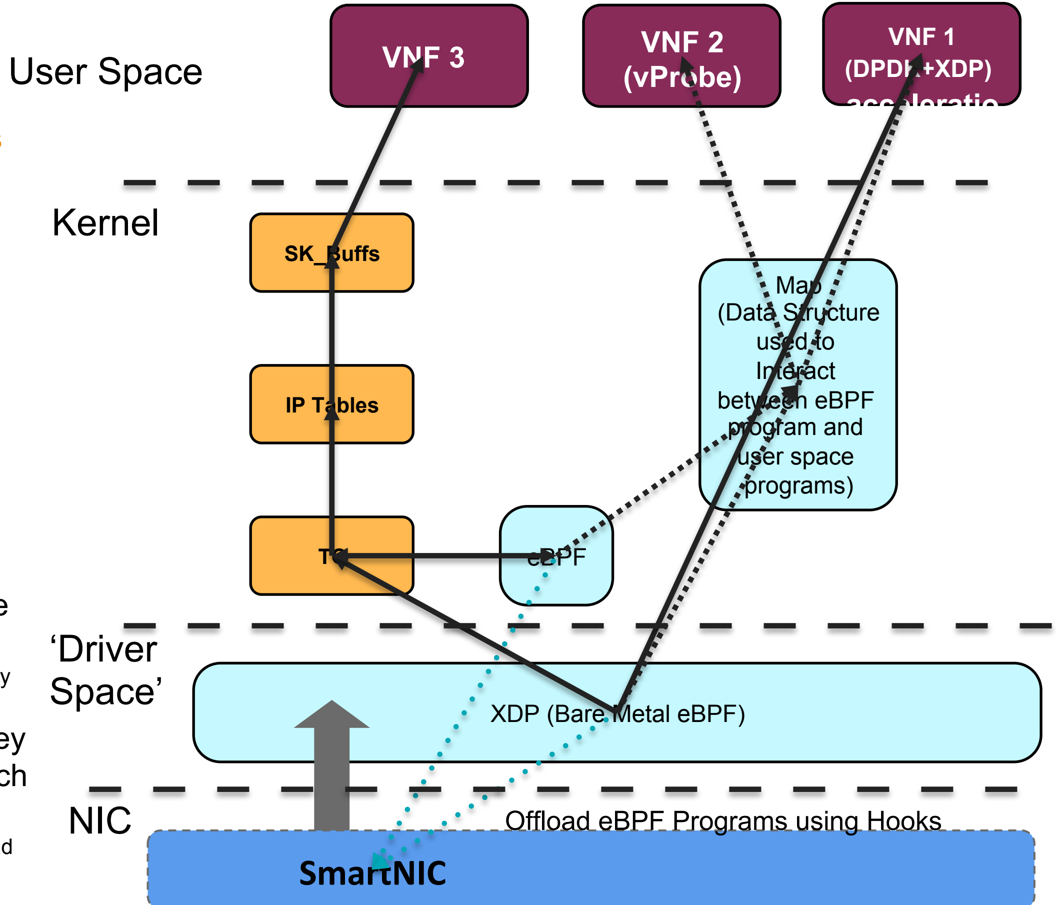
- OVS/vRouter is a great example of the benefits of Acceleration as a service
  - Kernel OVS < 10 Mpps using ~10 cores for processing
  - DPDK ~12 Mpps using 10-14 for processing
  - SmartNIC ~30Mpps using 1 core (NFP)
- By offloading the fast path, but maintaining fallback to kernel and Userspace offload ensures transparency
  - Can be used with various NIC vendors
- Finally, if offload is removed, the OVS will simply run on the kernel
  - This means acceleration vendors will never be able to ‘trap’ their customers



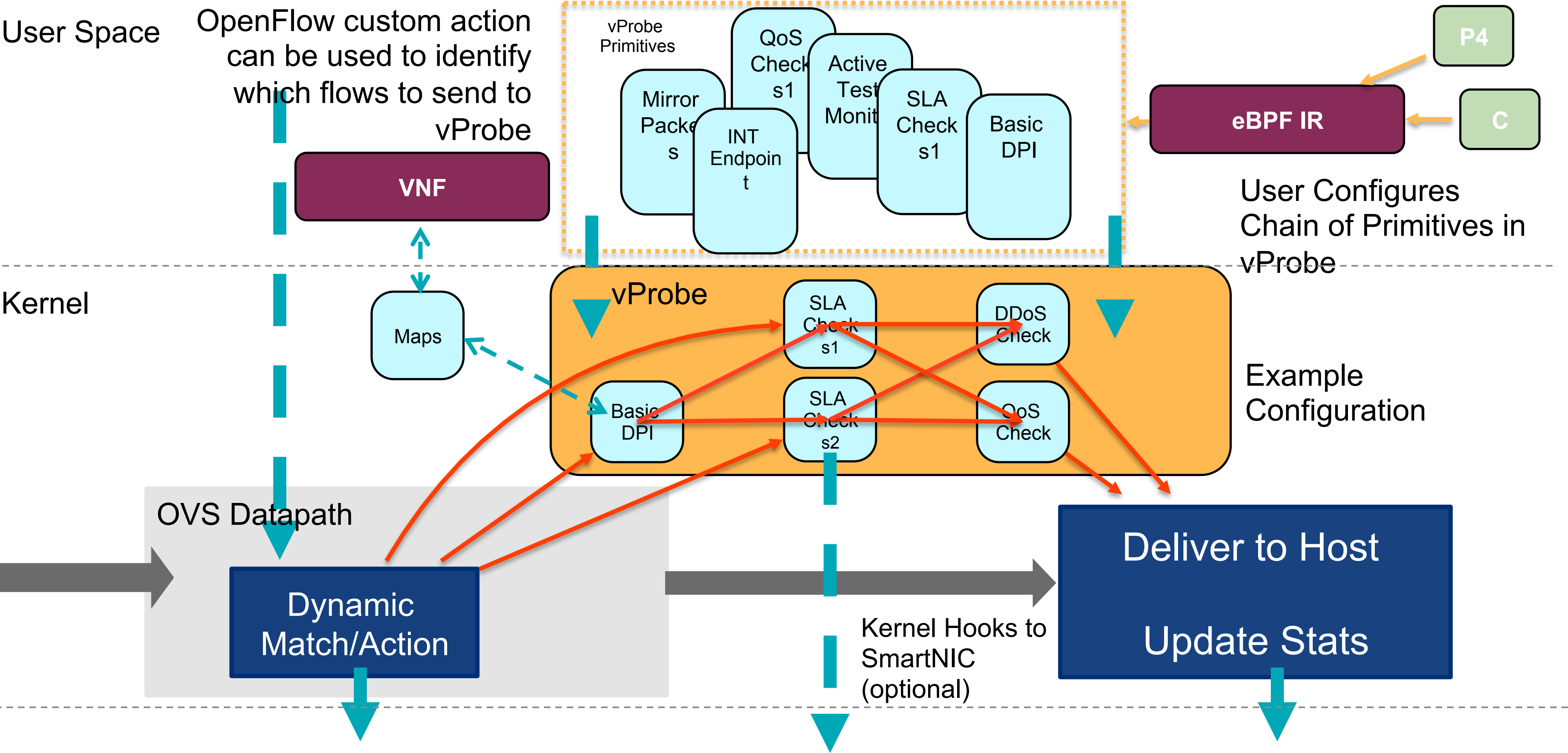
General Software Acceleration-eBPF/XDP

- There already exists a framework within the Kernel to accelerate software
- The development of the XDP variant of eBPF now means this is compatible with DPDK and other bypass infrastructures as well as the kernel data path

- eBPF programs are small contained C programs
  - 10 64 bit registers, 512 byte stack, infinite map storage
  - Limited jump backs-no infinite loops
  - 4K instructions
  - can be chained using tail calls with other eBPF programs
- These were previously limited for use with the traffic classifier in the kernel
  - TC provides simple match/action framework to filter packets and use eBPF programs
  - However only available if using kernel stack (i.e not DPDK)
- However the advent of XDP (eXpress Data Path) is aiming to enable these programs to be deployed more flexibly
  - Using 'driver space'/bare metal to host eBPF programs means that they can be used with DPDK etc.
- This then allows VNF providers a method whereby they can produce programs that are high performance which can be transparently accelerated
  - Thereby we have achieved our aim of totally transparent general offload
  - Also means the through **trusted executions environments (TEE)** carriers will be able to drive their own innovation


User Space
VNF 3
VNF 2 (vProbe)
VNF 1 (DPDK+XDP) acceleratio
Kernel
SK_Buffs
IP Tables
TC
eBPF
Map (Data Structure used to Interact between eBPF program and user space programs)
'Driver Space'
XDP (Bare Metal eBPF)
NIC
Offload eBPF Programs using Hooks
SmartNIC

Pulling it all together-Programming Model



Summary

- Using the eBPF/XDP model of general acceleration ensures cross platform acceleration that enables both vendor independence and performance
  - Ensures full exploitation of hardware
  - Ensures software/hardware combinations can be chosen through ‘plug and play’ approach
  - Opens up new opportunities for innovation in hardware and software
  - Reduces the risk taken by end users when experimenting with innovative solutions
  - Enables an increase in the speed of end user experience improvement