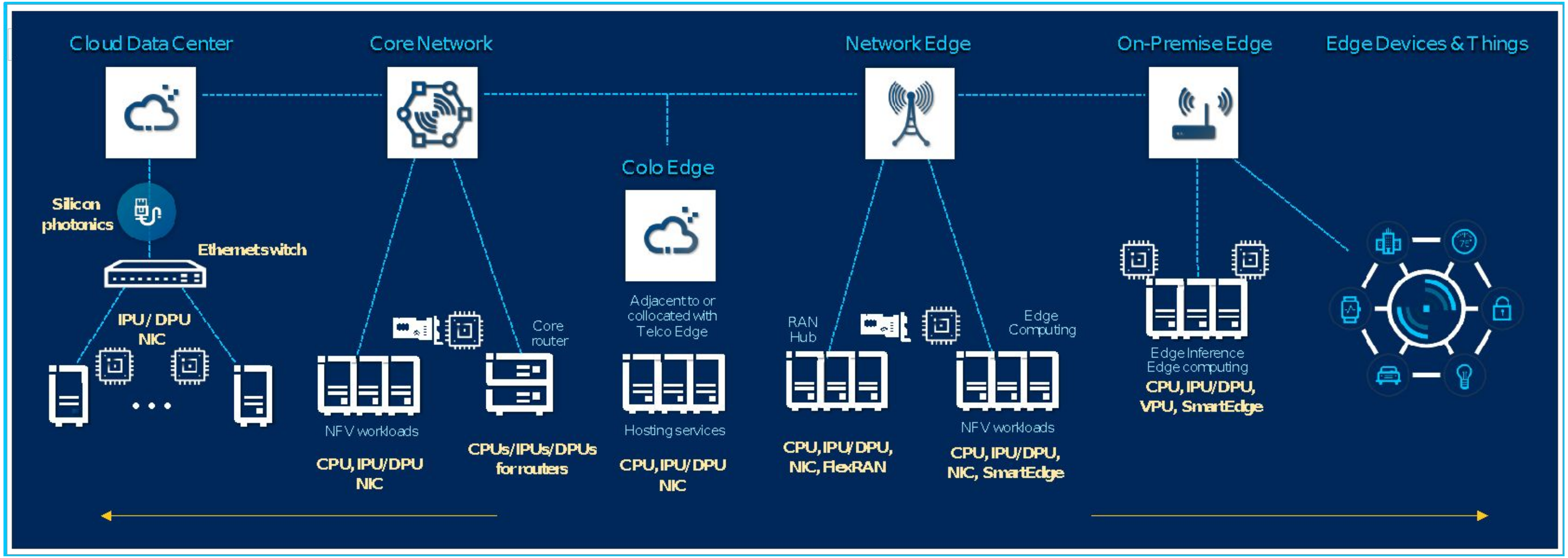
Breakout Session:

Target Abstraction Composition

Discussion on supporting programmable target abstractions

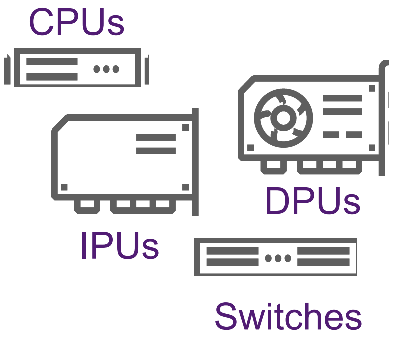
Nadav Haklai (Marvell)
Liron Himi (Marvell)
Dan Daly (Intel)

Need for Programmable Infrastructure



Industry Challenges

Lack of
Agreed Upon &
Effective Standards



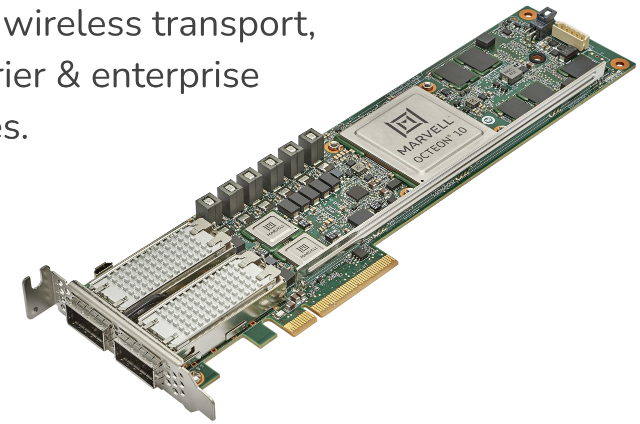
New Devices &
Hardware Emerging
Needing Programming

Motivating Example

Marvell

Marvell's OCTEON DPU family introduction

- OCTEON 10 DPU
  - Optimized for challenging hyperscale cloud workloads, 5G wireless transport, 5G RAN intelligent controller (RIC) & edge inferencing, carrier & enterprise data center applications and fanless networking edge boxes.
- OCTEON TX2 DPUs
  - 64-bit ARM SoC processors that combine up to 36 cores with configurable and programmable hardware accelerator blocks that support up to 200G data paths.
- OCTEON Fusion DPUs
  - CNF95XX family of DPUs provide 5G and LTE-A PHY layer (L1) functionality, supporting base station designs ranging from high-capacity micro cells to multi-sector macro cells. CNF95XX also support O-RAN and vRAN distributed units.



Marvell's DPU Software Standardization Efforts

- Active participation in OPI subgroups – Vision, API, PoC, etc.
- Dataplane Workload Accelerator library (DWA) - DPDK RFC
- OpenShift "DPU mode" integration
- IPDK project - Evaluation and contribution

RED HAT OPENSHIFT

ipdk Infrastructure Programmer Development Kit

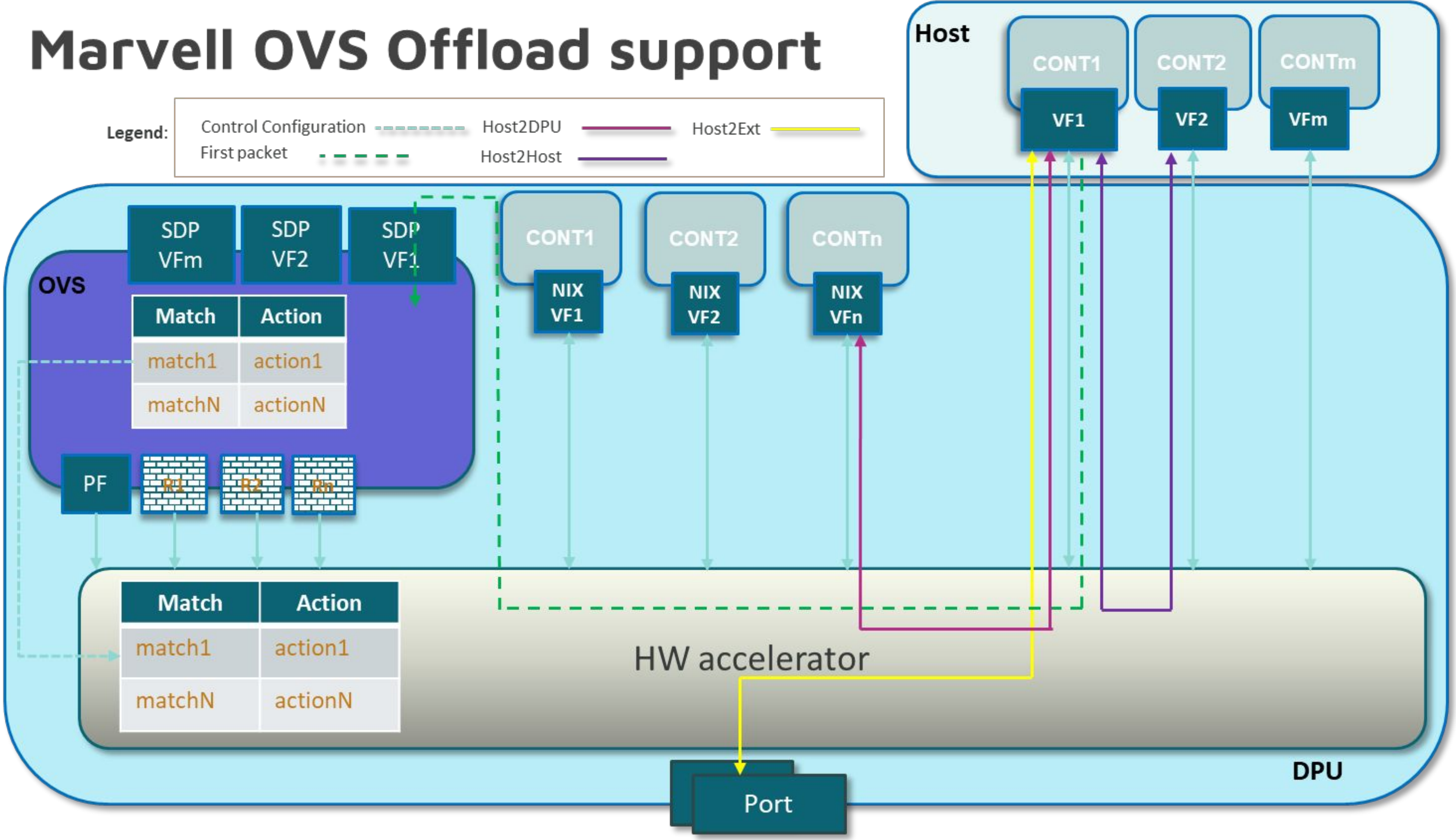
DPDK DATA PLANE DEVELOPMENT KIT

Marvell IPDK Evaluation

Marvell OVS Offload support

- OVS running on the DPU’s cores
- Offload - ARM CPU / Hardware accelerators
- Containers on Host and DPU with full connectivity
- Service chaining

Marvell OVS Offload support

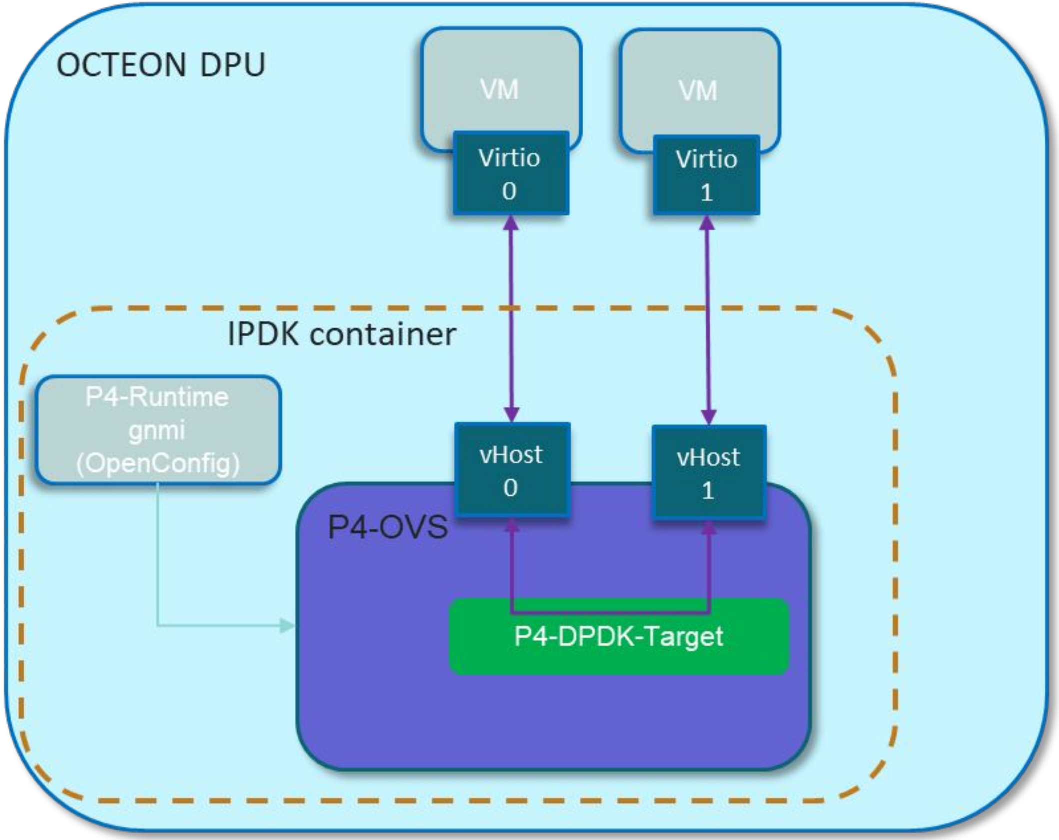


Marvell’s DPU PoC - IPDK exploration

- Motivation:
  - Explore p4 based solutions for OCTEON DPU family
  - IPDK PoC on a OCTEON ARM based platform
- PoC goals:
  - Install and run IPDK with VMs running on OCTEON DPU
  - Enable IPDK on OCTEON DPU with VMs running on Host
  - Basic traffic tests with p4 DPDK target
  - Performance benchmarking

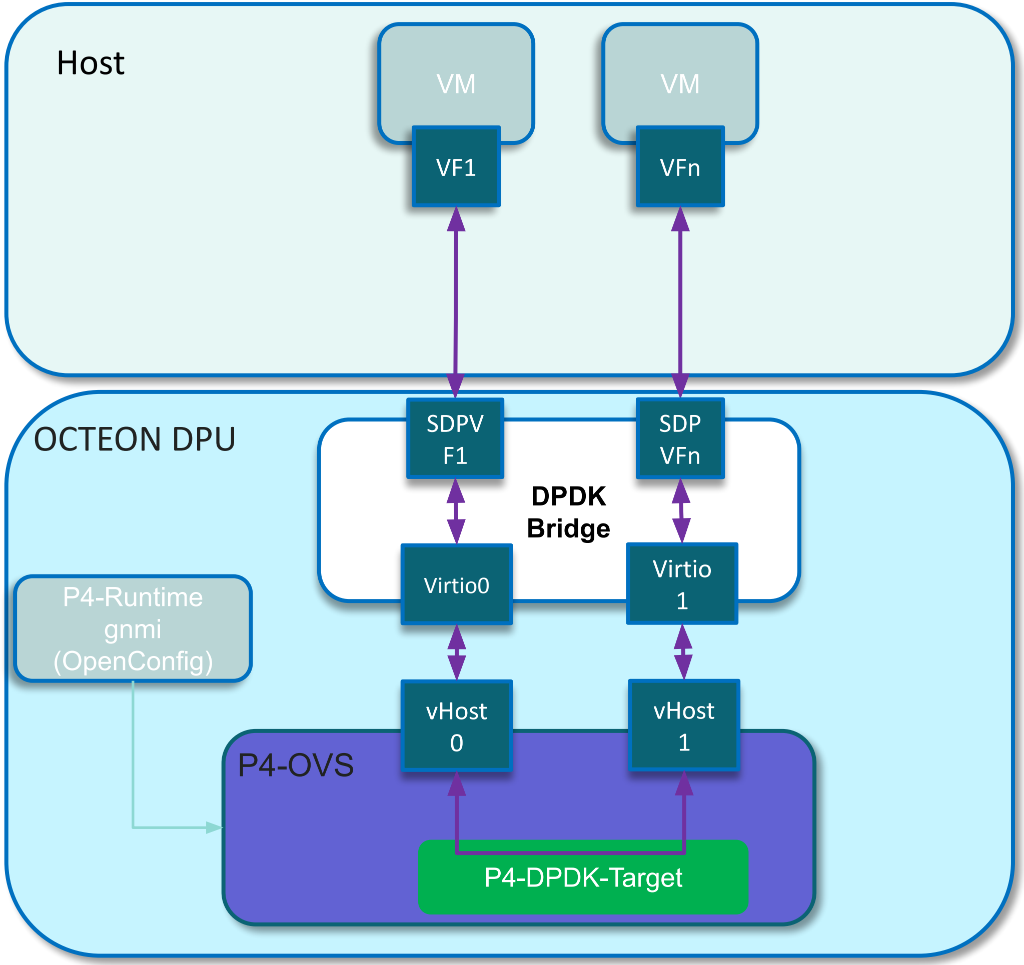
IPDK on OCTEON with VMs on DPU

- Connect VMs to the virtio interfaces
- L3 sample PSA P4 based app
- Using P4-DPDK-Target
- Solved ipdk ARM integration issues



IPDK on OCTEON with VMs on Host

- Host: Attach the OCTEON Host VFs to VMs
- DPU: Forward traffic between the p4-ovs virtio interfaces and SDP VFs
- Performance testing between Host VMs



* SDP – Host VF representation

Marvell OCTEON IPDK PoC conclusion

- IPDK working smoothly on OCTEON DPU
  - ARM support was missing - added and upstreamed by Marvell
- p4 DPDK target
  - Performance limitations - CPU Scalability
- PCI Interface support missing
  - Virtio only
  - Required for external interface
  - Required for DPU->Host interface

How to Abstract Targets?

Requirements:

- Standards Based
- Export Full Functionality
- Support the full range of vendors & implementations

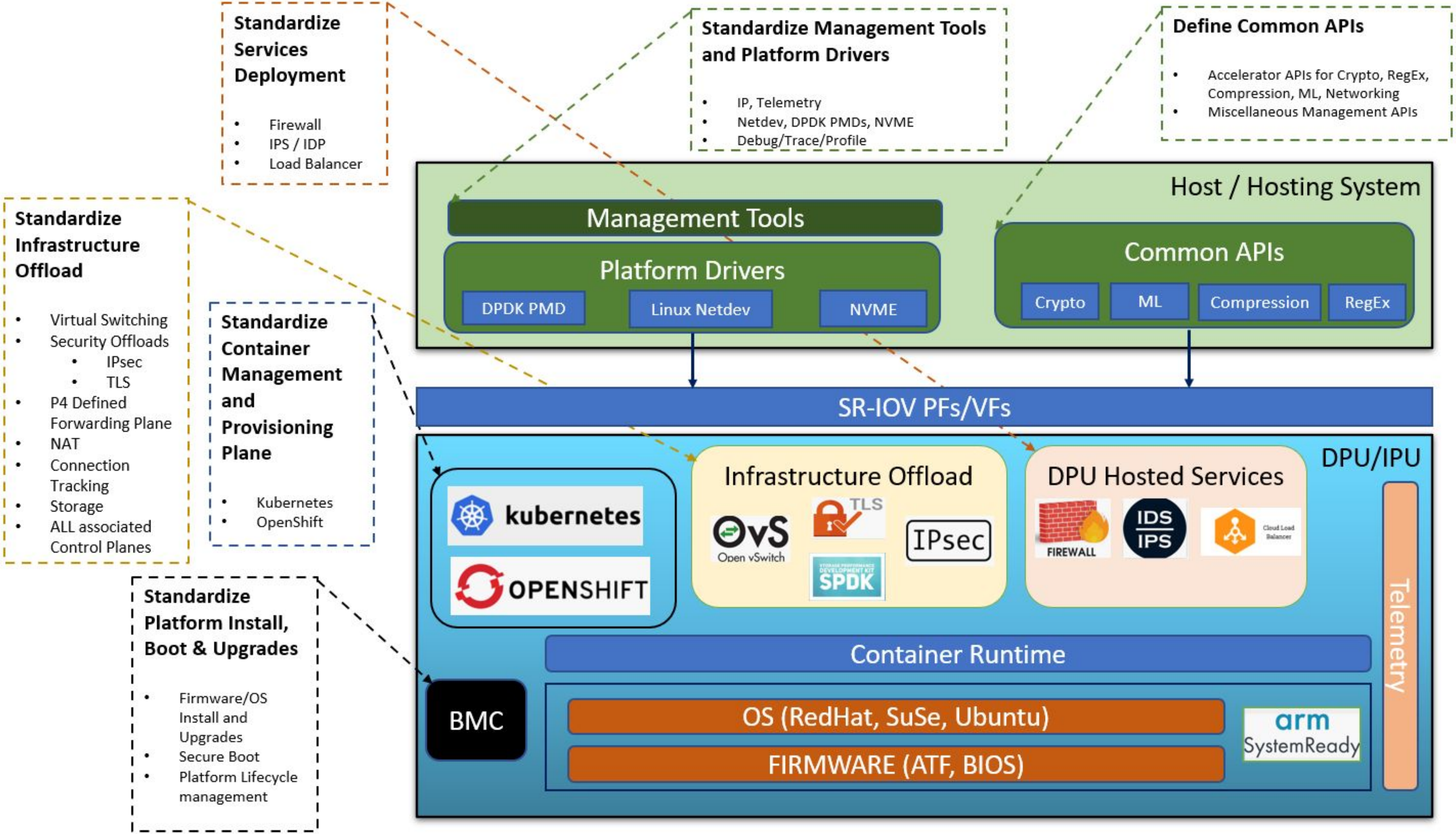
No Obfuscation

No Favorites

Software Standardization Vision

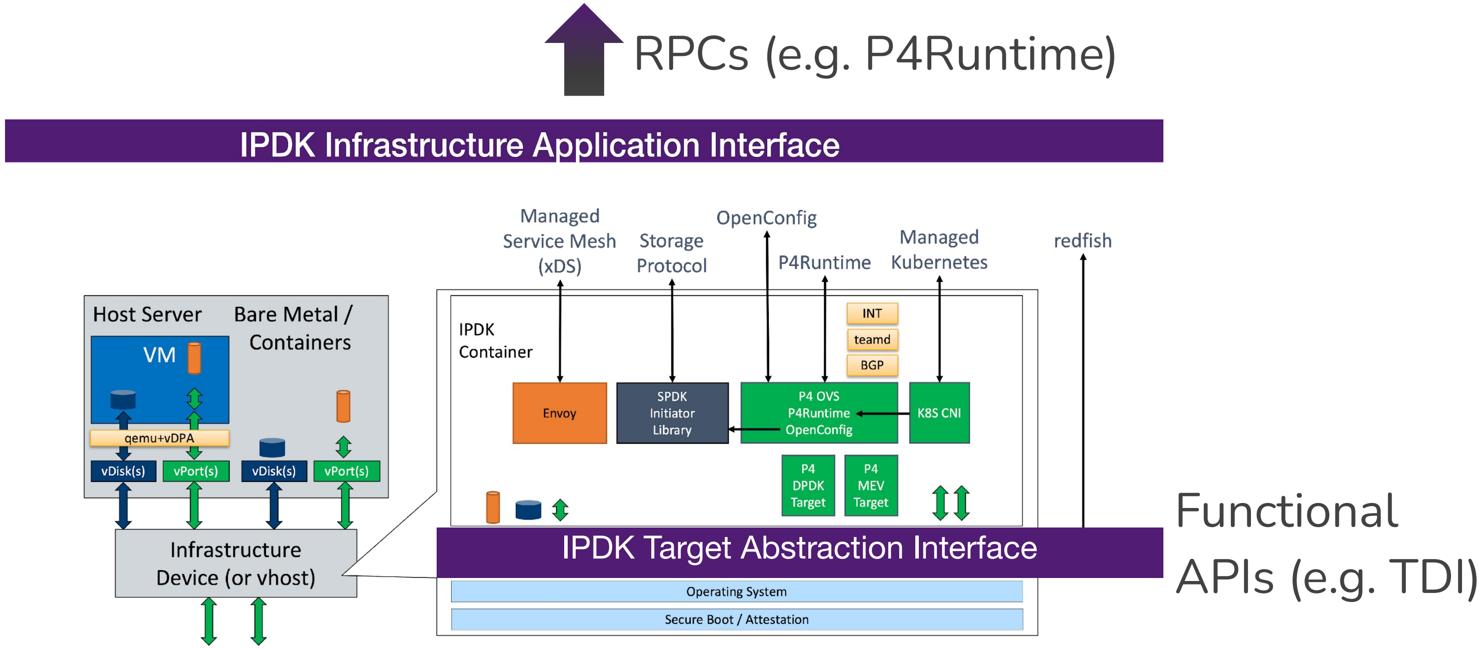
Define Vendor/Platform independent mechanisms for discovery, configuration and offloading/scheduling of workloads to DPUs

- Standardize common DPU/IPU features
  - Standardize Management Tools and Platform Drivers, FW updates etc.
  - Infrastructure Offload and Workload Deployment
  - Container Management and Provisioning Plane
- Expose unique features and capabilities in a standard way
  - Define Common APIs (Accelerator and other)
  - For example - TLS, IPsec functionality



Standardize Services Deployment
Firewall
IPS / IDP
Load Balancer
Standardize Management Tools and Platform Drivers
IP, Telemetry
Netdev, DPDK PMDs, NVME
Debug/Trace/Profile
Define Common APIs
Accelerator APIs for Crypto, RegEx, Compression, ML, Networking
Miscellaneous Management APIs
Standardize Infrastructure Offload
Virtual Switching
Security Offloads
IPsec
TLS
P4 Defined Forwarding Plane
NAT
Connection Tracking
Storage
ALL associated Control Planes
Standardize Container Management and Provisioning Plane
Kubernetes
OpenShift
Standardize Platform Install, Boot & Upgrades
Firmware/OS Install and Upgrades
Secure Boot
Platform Lifecycle management
Host / Hosting System
Management Tools
Platform Drivers
DPDK PMD
Linux Netdev
NVME
Common APIs
Crypto
ML
Compression
RegEx
SR-IOV PFs/VFs
DPU/IPU
kubernetes
OPENSHIFT
Infrastructure Offload
TLS
OvS
Open vSwitch
IPsec
SPDK
DPU Hosted Services
FIREWALL
IDS
IPS
Telemetry
Container Runtime
BMC
OS (RedHat, SuSe, Ubuntu)
FIRMWARE (ATF, BIOS)
arm
SystemReady

Example: IPDK Interfaces



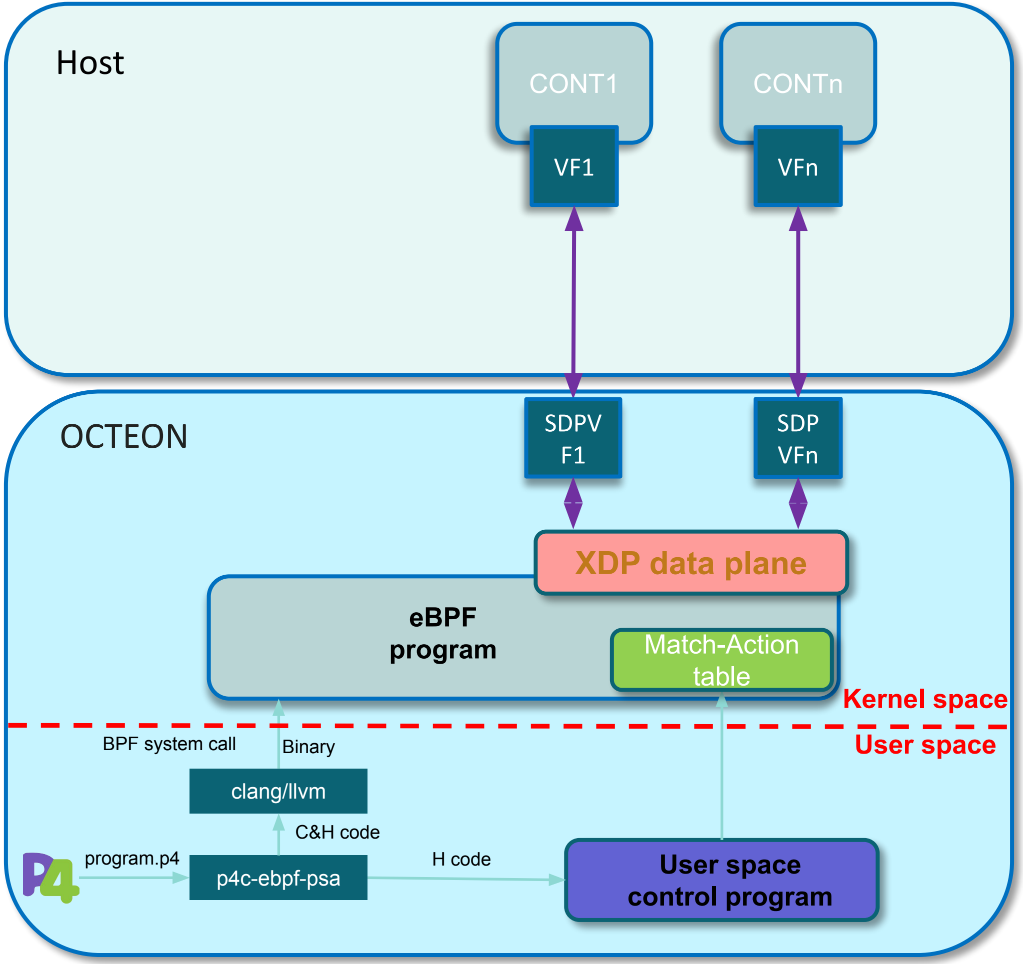
How Do We Get Organized?

Q&A

We could not cover all details, so ask away

XDP Datapath

- Try improve p4 DPDK target performance with XDP
- Test:
  - P4 program (l3fwd) compiled to eBPF program.
  - Hook the eBPF to the SDP VFs (kernel interfaces)
  - Forward traffic between the VFs

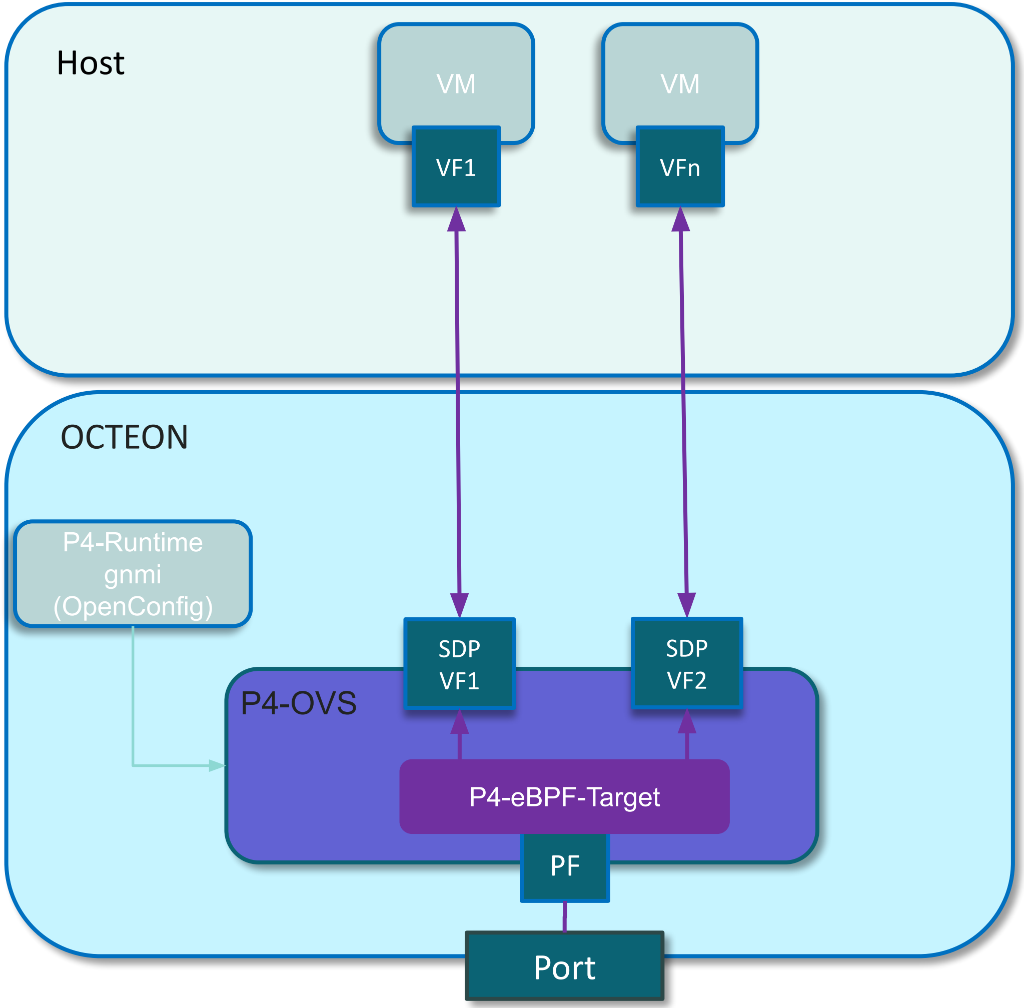


p4 target Datapath performance summary

- P4 DPDK target
  - No CPU scalability (No RSS support)
  - DPDK SWX pipeline performance limitations:
    - Single core per p4 pipeline
    - Scalability can be achieved with RSS and duplicated p4 pipelines
  - IPDK's default
- eBPF/XDP
  - Linear CPU scalability
  - IPDK eBPF – not yet supported. In roadmap
- **Performance comparison (p4-XDP vs p4-DPDK) – x12**
- Marvell p4 target – To be explored

IPDK OCTEON - Gaps

- eBPF support
- PCI interfaces support



THANK YOU!