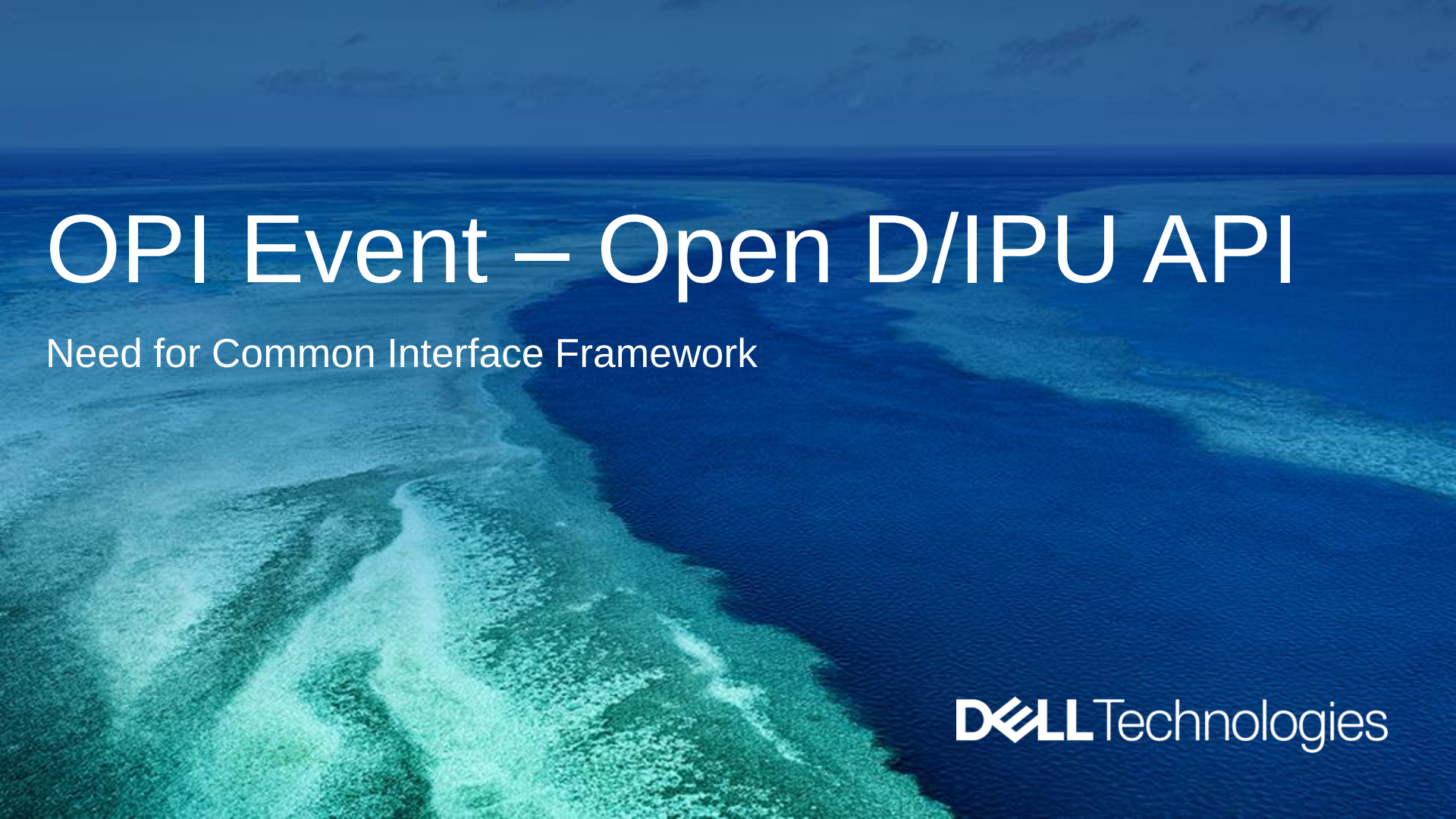
OPI Event – Open D/IPU API

Need for Common Interface Framework



Multiple Vendors

Solutions and Ecosystems

- Vendors
  - NVIDIA
  - Pensando
  - Intel
  - Marvell
  - Xsight
  - Fungible
  - AMD/Xilinx
  - …
- Cards and Silicon
  - Bluefield, Bluefield2
  - DSC-25/100, DSC-200
  - Mt Evans
  - CN9xxx
  - Octeon 10
  - FC50, FC100, FC200
  - …
- Ecosystems
  - DOCA
  - IPDK
  - Various Marketplaces

DPU/IPU Roulette

Multiple Behaviors, Multiple Interfaces, Multiple Frameworks

- High Level Behavioral Models
  - SDXI (SNIA)
  - DASH
  - Redfish (DMTS)
  - OpenBMC
  - OpenConfig
- Internal Vendor System Level
  - DOCA
  - IPDK
- Low Level
  - Vendor SDKs
  - Pipeline interfaces
  - DPDK
  - SPDK
  - RegEx
- Operating Environment (Linux)
  - Container
  - OS Acceleration (eBPF)
- Service Specific
  - Networking
  - Storage
  - Security
  - Gateway
  - Accelerator (AI/ML)
  - Telemetry
- Lifecycle Management

- Need a community agnostic API set to provide a common behavioral interface set for the DPU/IPU multi-vendor ecosystem

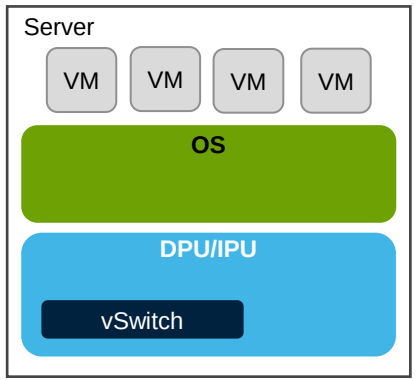
DELL Technologies

Key Use Cases for D/IPU

0 ***Better / Distributed Security (cross-cutting use-case)***

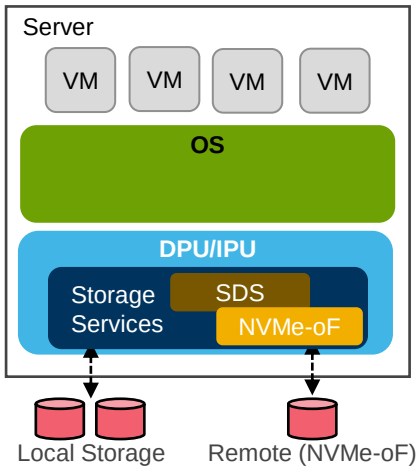
1 Network Offload and Disaggregation

Virtualize the ToR and offload network functions



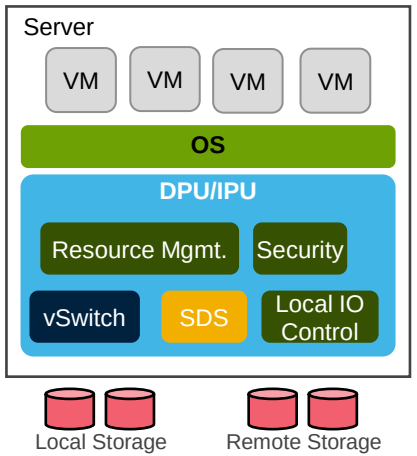
2 Storage Offload and Disaggregation

Offload storage functions



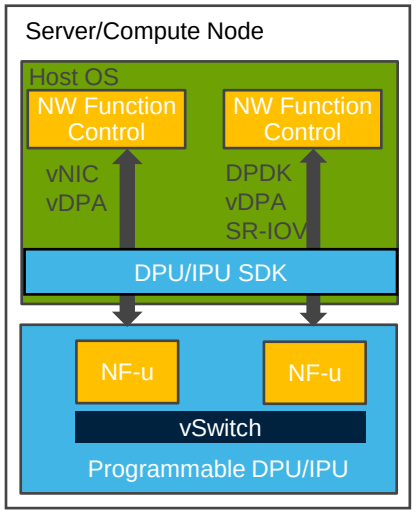
3 Bare Metal

Utilize as a Bare Metal Controller



4 Edge/5G

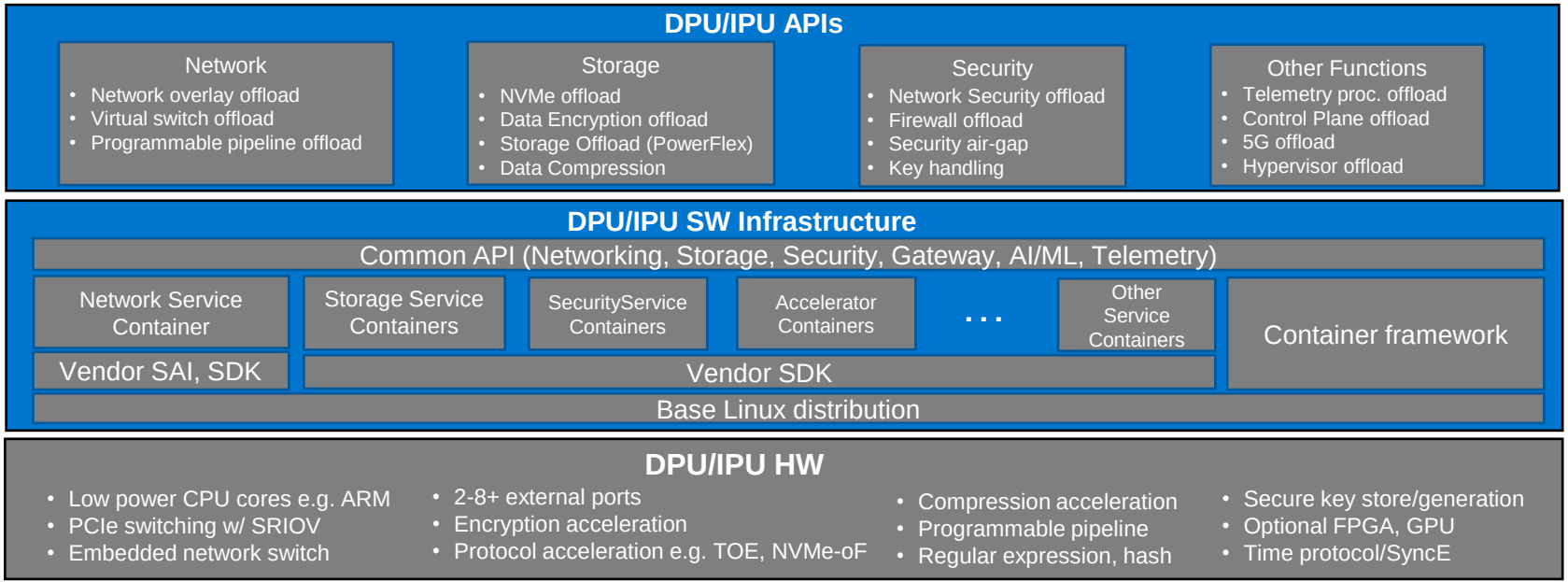
Offload I/O intensive functions



Need for an Open API for D/IPU

- Define standard mechanisms for Service Deployment
- Support of a Multi-Vendor Open D/IPU API definition and adoption for
  - Storage Services
  - Network Services
  - Security Services
  - AI/ML
  - Telemetry
  - System and Lifecycle Management
- Reuse Existing or define new common APIs for Configuration, Management and Consumption

Top Level Framework View



API Scope

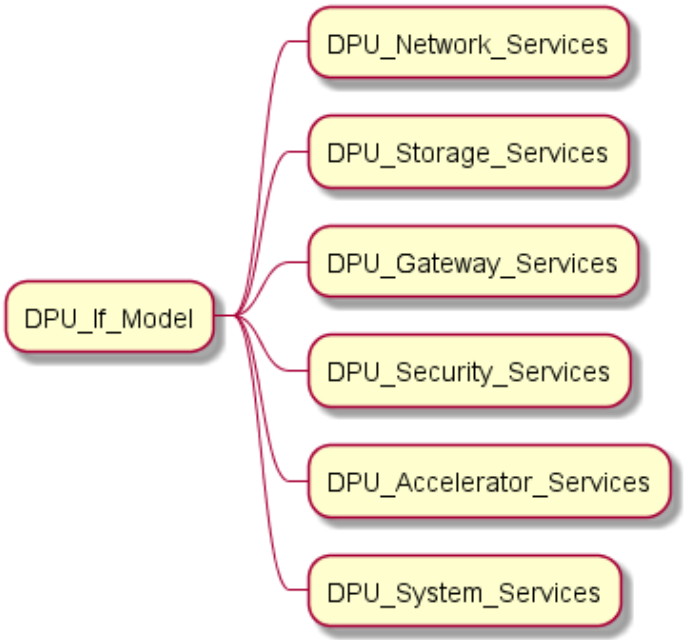
- System
  - Systems Management & Lifecycle
    - (Redfish)(OpenBMC?)(etc)
  - Monitoring, Metering, & Telemetry
- Operating System (Linux)
  - Standard Linux Libraries and packages
  - Container and Application Hosting
- Hardware (PCIe…)
  - Virtual Function Mapping
  - Offload Configuration
- Low Level APIs
  - Micro-Code in Data Flow Processing Cores
  - P4 Packet Processing Pipelines
  - Leverage commonly used APIs
    - DPDK, SPDK, EBPF
- Vendor Unique API & SDK
  - *These are NOT common/Open APIs*
  - DOCA, ASAP2, SNAP

- Storage
  - Networked Storage
    - NVMe/TCP
    - NMVe/RoCE(RDMA)
  - Storage Services
    - RAID/Erasure Coding/etc
  - Compression
  - SDXI Offload
- Networking
  - SONiC
    - OpenConfig (includes BGP, etc)
    - SAI implementation by the DPU
  - Policing and QoS and SLA
  - Multi-tenant Overlay
  - Host facing NIC Configurations
  - OVS
- Gateway
  - Connection Tracking
  - Load Balancing
  - NAT
  - Tunnels
- Security
  - Policy & Filters
  - Crypto Offloads
  - Secure Storage
    - keys, secrets, attestation, ...
  - Key Management
  - Network security offload
    - (TLS, IPSec)
  - RegEx matching

- Other Groupings such as Accelerators, AI/ML, etc.

DELL Technologies

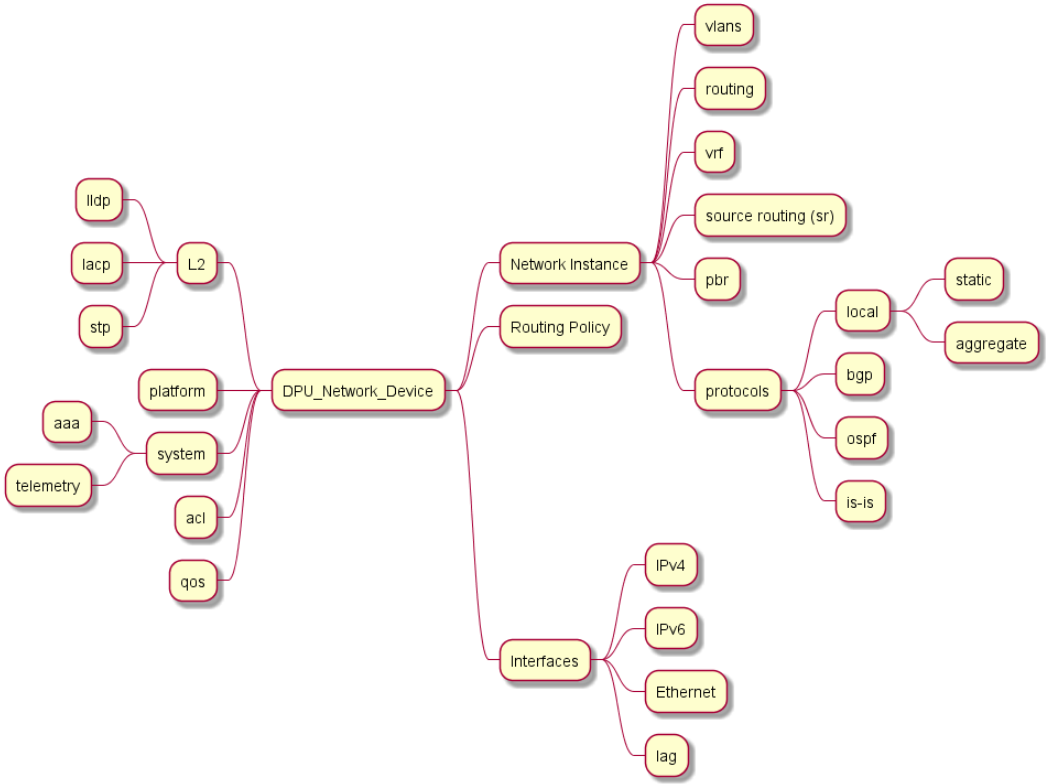
Open Programmable Infrastructure API Model



- Conceptual initial view of the API breakdown of services available in the D/IPU platforms
- Ability to support service chaining to provide the desired operation

Network Services API

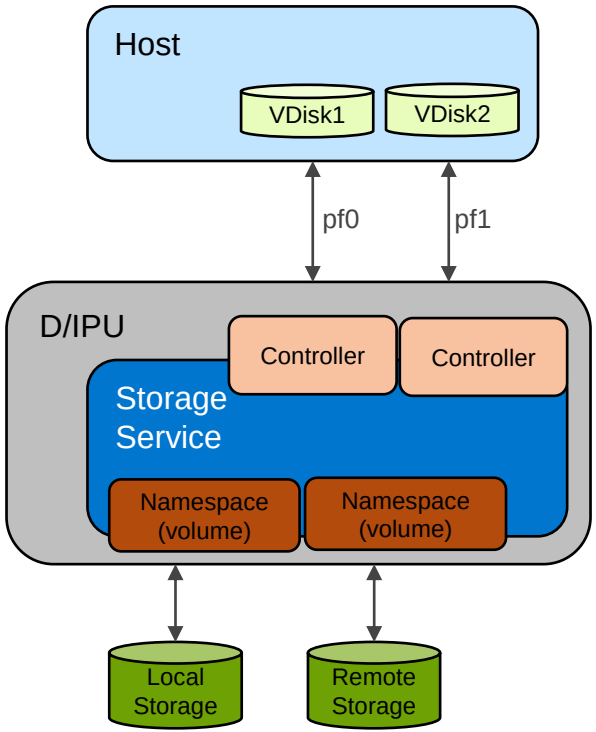
- Utilize the OpenConfig model for network service configuration
- Aligns with many Networking OS environments for configuration and management



Storage Service Example

High Level View (One of Many)

- Considerations for Storage API
  1. Setup Network Interface
     - IP Address (IPv4/IPv6), QoS, VxLAN, etc
  2. Create NVMe Subsystem (optional)
  3. Create the Controller
     - For each PF/VF
  4. Create the Namespace for the local or remote storage
     - PCIe, RDMA, TCP
  5. Attach the Namespace to the Controller




DELL Technologies