Open Programmable Infrastructure
Project Introduction

How we can together implement
DPU/IPU Infrastructure across all Vendors

Dr Joseph L White OPI TSC Chair & Dell Fellow/VP
Abstract

The Open Programmable Infrastructure (or OPI) is an open-source effort within the Linux Foundation to develop a standard API for utilizing SmartNICs, DPUs and IPUs, and other coprocessors or processing elements. It will allow users to provision and orchestrate all devices in the same way, thus allowing them to handle many different devices, implement new devices, and change or replace devices without learning a new command structure. It will also allow manufacturers to create a standard API, deliver new or upgraded devices faster, and benefit from a large ecosystem. It makes learning curves for new devices shorter and implementation or software errors easier to find. It opens new markets for devices and eliminates concerns over one-of-a-kind implementations.

This session will explore the goals and progress that the OPI project has undertaken. DPUs have many different use cases implemented by many different vendors. Our goal is to define a common framework for all of these devices to meet those use cases: This includes Infrastructure/ workload isolation, Security, Network offload and acceleration, and Storage offload and acceleration. Over the last year since OPI joined the Linux foundation we have welcomed 14 member companies to our project spanning the landscape from vendors, to integrators, to test infrastructure vendors, end users, as well as operating system and ISV vendors.

We have sought to create common provisioning and lifecycle management frameworks, defined APIs for the management of these devices to meet the most common use cases we have researched from end users and developed a developer platform and lab to test and explore these common frameworks.

Come listen to industry experts as we explore the DPU/IPU ecosystem and the OPI project’s progress toward a common set of frameworks, and how these assist end users with ease of deployment, lowers the total cost of development and ownership, and thus provides for broader adoption of this new class of devices.
Why should you care about OPI for DPU/IPU?

- With Moore’s law slowing down, ever increasing demands for compute, and exponential growth in data traffic…
  - We need heterogeneous compute
  - We need composability.
  - Workload specific resources per host
- Hyperscalers deploy DPU/IPUs w/ non-standard frameworks
  - We want Standard APIs for Edge, Telco, Enterprise
- Hardware needs to be abstracted
  - Solution providers can focus on deploying services
  - Ease of development & deployment
- Need to drive efficiency in large computing environments > TCO savings
- Standards and common APIs needed to drive broader adoption of DPU/IPUs
  - Flywheel effect
Project Goals

- Create community-driven standards-based open ecosystem for DPU/IPU-like technologies
- Create vendor agnostic framework and architecture for DPU/IPU-based software stacks
- Reuse existing or define a set of new common APIs for DPU/IPU-like technologies when required
- Provide implementation examples to validate the architectures/APIs
New chapter in modern system architecture

**Traditional SmartNIC model**

- Computer is CPU + SmartNIC as peripheral that is fully controlled by the CPU
- CPU + domain-specific HW acceleration
- Static device function

**DPU/IPU model**

- NIC & HW accelerators move to DPU/IPU-like device with its own CPU
- Software defined device function
- Computer is an aggregation of independently intelligent subsystems
DPU Example Expanded

Hosting System

- 4-32 or more Cores plus DRAM & HBM
- Protocol Acceleration: NVMe-oF, TCP, RoCE, P4 Pipeline
- Inline micro-processors
- Encryption Secure Keys
- RegEx
- Hashing

Storage
GPU
CPU
Memory

BMC

DPU

PCIe
General Cores
Accelerators

Compression
Encryption
GPU, AI Processing

Storage Controller
Regular Expression
FPGA

P4 Pipelines

Encryption
Secure Keys
RegEx
Hashing

Protocol Acceleration
NVMe-oF, TCP, RoCE
P4 Pipeline
Inline micro-processors

2-8 Ports
Embedded Switching
Network Packet Processing
NIC Functions

mgt

Switch

OPEN PROGRAMMABLE INFRASTRUCTURE PROJECT

8 ©2023 SNIA. All Rights Reserved.
DPU / IPU Use cases

**Infrastructure workload isolation**
- Control Plane offload
- Host Lifecycle and Provisioning
- Host Offloads

**Security**
- Security domains (Host and DPU/IPU)
- FW, intrusion detection and prevention

**Networking offload and acceleration**
- Virtual switch offload, IPSEC, TLS

**Storage offload and acceleration**
- nvme/tcp offload, compression, and dedupe acceleration

**Applies Across:**
- Edge
- Enterprise
- Cloud
- Telco Core & Edge
OPI Organizational Structure

- Board of Directors
- Outreach Committee
- Technical Steering Committee

Provisioning & Lifecycle  API & Behavioral Model
Developer Platform  Use Case

OPI Technical Deliverables
- Open-Source Projects
- Specifications/Standards
- Reference Platforms
- Test Suites & Cases
- POC/Prototypes
Scope and Goals of Working Groups

Provisioning & Lifecycle
- Discovery & Provisioning
- Inventory
- Boot sequencing
- Lifecycle & Updates
- Monitoring & Telemetry

API & Behavioral Model
- Object models
- Host & Management facing APIs
- Taxonomy for Services (Networking, Storage, Security)
- Re-use industry standard APIs (OpenConfig, VPP, FRR, etc)
- Reference Orchestration Client

Developer Platform
- Independent testing Lab
- Virtual & Hardware POCs
- Simulation Environment
- CI/CD

Use Case
- Areas of high interest
  - Storage, Security, Networking, AI/ML
- Use cases gathered from end users
  - OVS/OVN
  - NVMe/PCIe to NVMe/TCP bridge
  - Basic Firewall.
# OPI Repositories

<table>
<thead>
<tr>
<th><strong>opi</strong></th>
<th>OPI Main Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>opi-prov-life</strong></td>
<td>Provisioning, Lifecycle and Platform Management</td>
</tr>
<tr>
<td>- Discovery &amp; Provisioning</td>
<td></td>
</tr>
<tr>
<td>- Inventory</td>
<td></td>
</tr>
<tr>
<td>- Boot sequencing</td>
<td></td>
</tr>
<tr>
<td>- Lifecycle &amp; Updates</td>
<td></td>
</tr>
<tr>
<td>- Monitoring &amp; Telemetry (OTEL)</td>
<td></td>
</tr>
</tbody>
</table>

| **opi-poc** | Developer Platform and PoC Work |
| - Integration Platform Definition |
| - Software Networking PoC via p4-ebpf |
| - spdk based storage device PoC |

| **opi-api** | Open Programmable Infrastructure API and Behavioral Model |
| - Create a Taxonomy for services: |
| - Networking |
| - Security |
| - Storage |
| - Gateway |
| - Telemetry |
| - AI/ML |

## SPDK
- **opi-spdk-bridge**
  - OPI Storage gRPC to SPDK json-rpc bridge POC
- **opi-nvidia-bridge**
  - OPI gRPC to Nvidia bridge third party repo
- **opi-marvell-bridge**
  - OPI gRPC to Marvell bridge third party repo
- **opi-spdk-bridge**
  - OPI storage gRPC to SPDK json-rpc bridge
- **spdk-csi** (Forked from spdk/spdk-csi) & spdk
  - CSI driver to bring SPDK to Kubernetes storage through NVMe-oF or iSCSI. Supports dynamic volume provisioning and enables Pods to use SPDK storage transparently.

## godpu
A Container Storage Interface (CSI) library, client, and other helpful utilities created with Go for OPI

## pydpu
Python library and cli to communicate with DPUs and IPUs

## sessionOffload
(Forked from att/sessionOffload)
Open API for IP Applications to Offload TCP/UDP Session Packet Processing to Hardware

## opi-strongswan-bridge
OPI IPSEC gRPC to strongSwan vici API bridge

## opi-smbios-bridge
OPI gRPC to SMBIOS bridge for inventory

## smbios-validation-tool
(Forked from google/smbios-validation-tool)

## sztp & sztpg
Secure Zero Touch Provisioning (sZTP) in OPI

## otel
Common DPU Telemetry definition

## opiproject.org
OPI Hugo Website

## artwork
OPI related logos and artwork.
Momentum and Progress

ACTIVE ORGANIZATION
- Dell Technologies
- Intel Corporation
- Red Hat, Inc.
- FS, Inc.
- Marvell Semiconductors Ltd
- Hewlett Packard Enterprise Development LP
- NVIDIA Corporation
- Arm Limited
- Keysight Technologies Inc.
- The Linux Foundation

TOTAL LANGUAGES
- Go
- Unknown programming language
- Java
- YAML
- AMPL, Linux Kernel Module, Module-2, XML
- Markdown
- Python
- Protocol Buffer
- C++
- C, C++, Objective-C
- Other Languages

TOTAL CONTRIBUTORS
- 162 \(\uparrow 296\%\)

TOTAL COMMITTS
- 4,442 \(\uparrow 2066\%\)
Create a Multi-Vendor Open API definition for

- Storage Services
- Network Services
- Security Services
- AI/ML
- Telemetry
- System and Lifecycle Management

Provide Industry standard interface for DPU/IPU

Consistent, standardized, protobuf based GRPC APIs for network and storage objects
Various orchestration systems could integrate with the APIs once for various DPU/IPU vendors
DPU/IPU vendors would write the shim/translation to underlying vendor specific SDK
Target use cases

Data center: On-prem cloud, public-cloud, 5G cloud
K8s clusters
Physical/Virtual appliance acceleration

Initially focus on Networking and Storage Interfaces for DPU/IPU

Multi-Tenant Shared Cloud
EVPN Gateway
IPSec (strongSwan)
Storage Initiator/Target

Provide open source prototype/reference implementation to the APIs
API Mechanism

**gRPC for configuration/control interface through API Gateway**

- Direct delivery of gRPC messages to appropriate shim layer
- gRPC to REST translation
- Support gNMI and gNOI

**Expose VF/PF for data consumption**

**Networking**
- Multi-tenant public cloud
- EVPN based telco cloud
- K8s based on-prem deployment

**Storage**
- NVMe Initiator/Target based storage virtualization

**Security**
- Strongswan based IPSec implementation
v1alpha API definition for storage solution
frontend (host facing), middle-end (services for volumes), and backend (target specific) APIs

Reference implementation ready
To map the OPI APIs to SPDK code running on the DPU/IPU
OPI SPDK Bridge https://github.com/opiproject/opi-spdk-bridge

Vendor support
Open source bridge code for AMD, Intel, Marvell, and Nvidia DPU/IPUs

CI/CD Integration
Continuous testing/validation against the reference implementation
Create a common API framework and extensions for
Cloud
Telco
K8s

Support network service capabilities
OVS, VPP, SONiC, …

Leverage existing API models
OVS, OpenConfig, CNI, …
Progress: Network API

v1alpha API definition for cloud
EVPN based telco-cloud use cases: https://github.com/opiproject/opi-api/tree/main/network/telco
IPSec on DPU/IPUs: https://github.com/opiproject/opi-api/tree/main/security

Reference implementations
Proposals to leverage open source to build reference implementations
IPSec with strongSwan https://github.com/opiproject/opi-strongswan-bridge

Vendor support
Cloud APIs: AMD
Telco Cloud APIs: Intel
IPsec Security APIs: Intel, Nvidia

CI/CD Integration
Continuous testing/validation on the API definition
APIs for implementing multi-tenant VPC (virtual private cloud)

- Tenant (VPC)
- Subnets/Networks
- Mapping (Endpoints)
- Routes
- Network Security Policies
- VPC Peering

Looking to making a reference implementation
OPI Networking API – EVPN Gateway
Demos

- **Networking Cloud Demo**
  - Video recording
  - API protocol buffers
  - Clients [https://github.com/opiproject/pydpu](https://github.com/opiproject/pydpu) and [https://github.com/opiproject/godpu](https://github.com/opiproject/godpu)

- **IPSec Demo**
  - Video recording
  - API protocol buffers
  - Clients [https://github.com/opiproject/pydpu](https://github.com/opiproject/pydpu) and [https://github.com/opiproject/godpu](https://github.com/opiproject/godpu)

- **Storage Demo**
  - Video recording TBD
  - API protocol buffers
  - Clients [https://github.com/opiproject/pydpu](https://github.com/opiproject/pydpu) and [https://github.com/opiproject/godpu](https://github.com/opiproject/godpu)
"Acceptance Test" for HW acceleration

Ability to measure and compare HW acceleration features
"Inbound" DPU, handle traffic inbound to host

Firewall and TLS offload
"Output" DPU, handle traffic from host

Switching and Loadbalancing, ECMP

Works well if Host is a proxy of some sort

"Hairpin" Host->DPU->Host pure HW acceleration test
No expensive optics or switching required!

Source->DPU->Host->DPU->Target is valid customer use case, but test is intended to be simpler
NGINX POC Example – F5 OPI Team

Traffic Generation.
- T-REX Open Source?
- Apache Bench?
- Maybe we work with Keysight?

Traffic Source Method

sZTP, TFTP, DHCP, DNS, etc services

X86 or ARM bare metal host.
- RH OpenShift
- NGINX Webserver

DPU Interface

VLAN 3 Mgmt

VLAN 1 External

VLAN 2 Internal

Enough NGINX webservers to Max out DPU throughput

VM or server. RH OpenShift installed and clustered with RH SNO on DPU

RH OpenShift

NGINX Webserver

VLAN 2

VLAN 1

Host Interface 1

PCIe bus

RHSNO

NGINX Proxy

Host Interface 1

Host Interface 1
The objective of the Open Programmable Infrastructure Project is to foster a community-driven standards-based open ecosystem for next generation architectures and frameworks based on DPU/IPU-like technologies.
OPEN PROGRAMMABLE INFRASTRUCTURE PROJECT

opiproject.org
Please take a moment to rate this session.

Your feedback is important to us.