SmartNICs and Data Processing Units (DPUs)

Computer Networks

Sina Daneshgar
Mohammadmohsen Abbaszadeh

Sharif University of Technology

Feb 2026

Outline

The Evolution of Compute

CPU General purpose, high flexibility, but high latency for simple repetitive tasks.

GPU Specialized for parallel processing (Graphics, AI).

DPU Specialized for **Data-Centric** tasks (Moving, Processing, Securing data).

Why now?

Moore's Law is slowing down, while networking speeds are exploding (10G $\rightarrow$ 100G $\rightarrow$ 400G). The CPU can no longer keep up with the networking interrupt load.

Quantifying the "Infrastructure Tax"

Data centers spend 20-30% of their total compute power just on "tax" tasks:

- **Virtualization**: OVS (Open vSwitch) overhead, encapsulation (VXLAN, Geneve).
- **Storage**: NVMe-oF (NVMe over Fabrics) protocol translation and management.
- **Security**: Distributed Firewalls, Micro-segmentation, and Wire-speed TLS/IPsec.

The Bottleneck

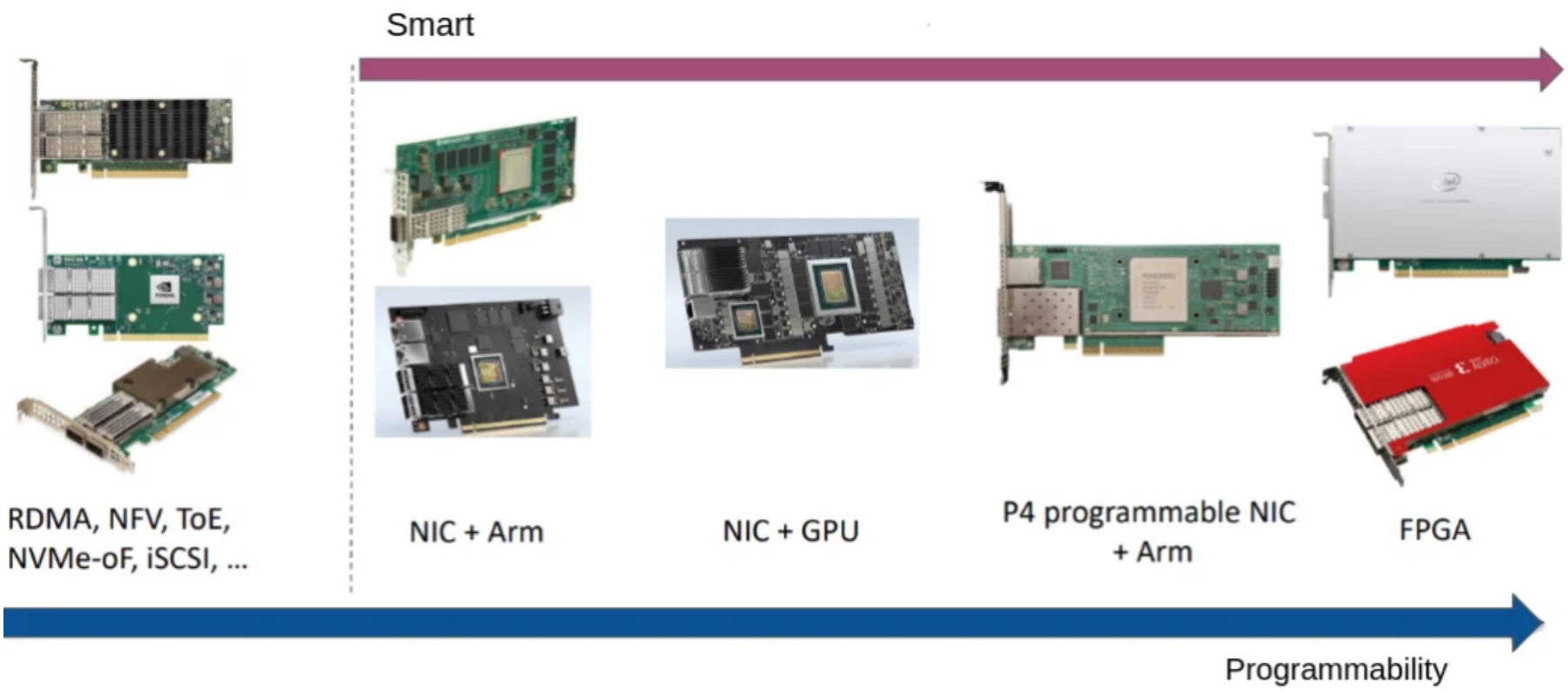
Every packet processed by the CPU is a cycle stolen from the user application (Application Stall).

Deep Dive: What's inside a DPU?

Unlike a standard NIC, a DPU (like NVIDIA BlueField or AMD Pensando) contains:

- **ARM/RISC-V Cluster**: Runs a standard Linux OS for management and control.
- **Network Acceleration Engine**: Programmable hardware for parsing and switching.
- **Hardware Engines**:
  - **Crypto**: Hardware-accelerated IPsec/TLS and Disk encryption.
  - **Storage**: VirtIO-blk acceleration and Compression engines.
  - **Timing**: PTP (Precision Time Protocol) for finance/telecom.
- **On-board RAM**: For local state, lookups, and buffering (16-32GB DDR4/5).

SmartNIC Landscape



SmartNIC landscape

- Spectrum from fixed-function NICs to highly-programmable DPUs (compare programmability vs. capability).
- Use this slide when you explain trade-offs: throughput, latency and power.

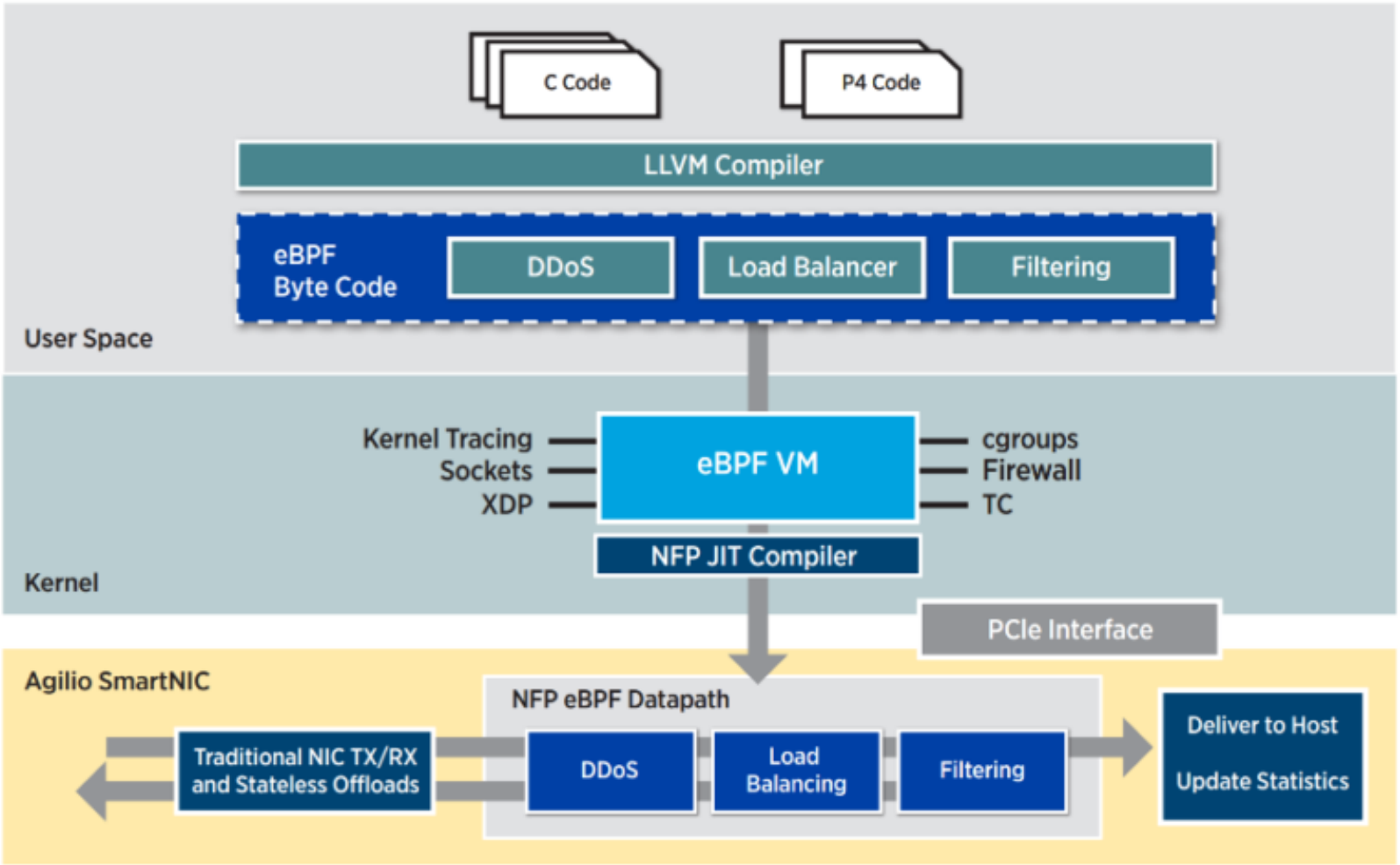
Programmability: P4 and the Match-Action Pipeline

- **P4 (Programming Protocol-independent Packet Processors)**:
  - Define custom headers and parsers.
  - Programmable Match-Action tables inside the NIC silicon.
  - Allows for "Protocol Independence."
- **Implementation**: See `examples/02-p4-programmability/basic_l2.p4` – a minimal L2 forwarding example (match-action table).

Programmability: eBPF and XDP

- **eBPF (Extended Berkeley Packet Filter)**:
  - Runs JIT-compiled bytecode inside the Linux kernel.
  - High safety (verified at load time) and extreme speed.
- **XDP (eXpress Data Path)**:
  - An eBPF hook placed as early as possible (the NIC driver).
  - Can DROP, FORWARD, or REDIRECT packets *before* the kernel allocates a socket buffer (`sk_buff`).
- **Demo**: See `examples/03-ebpf-xdp/03-ebpf_xdp.c` (ICMP-drop example).

eBPF / XDP: Runtime Architecture



- Kernel-level verification and JIT – safe, high-speed packet processing.
- XDP attaches at the driver level (earliest path) – ideal for filtering and DDoS mitigation.

Code: P4 – basic L2

```
// action forward sets egress port based on dst MAC
action forward(bit<9> port) { meta.egress_spec = port; }

// table: exact-match on dst MAC -> forward / drop
table mac_forwarding {
    key = { hdr.ethernet.dstAddr: exact; }
    actions = { forward; drop; }
}
```

- Simple match-action pipeline: lookups happen in hardware tables.
- Point to `basic_l2.p4` when explaining table installs and entries.

Code: eBPF/XDP – ICMP drop

```
if (ip->protocol == IPPROTO_ICMP) {
    // drop ICMP packets
    return XDP_DROP;
}
```

- Early drop in kernel driver path – zero-copy and minimal CPU.
- Attach with `ip link set dev <iface> xdp obj ...` (see file header).

Application 1: Cloud Resource Disaggregation

- **Traditional**: Storage and Compute locked in the same physical box.
- **DPU-based**: Storage is remote (over NVMe-oF), but the DPU makes it look like a local NVMe drive to the host OS.

Benefit

Storage performance is identical to local SSDs, but with the flexibility of network-attached storage.

Application 2: Bare Metal as a Service (BMaaS)

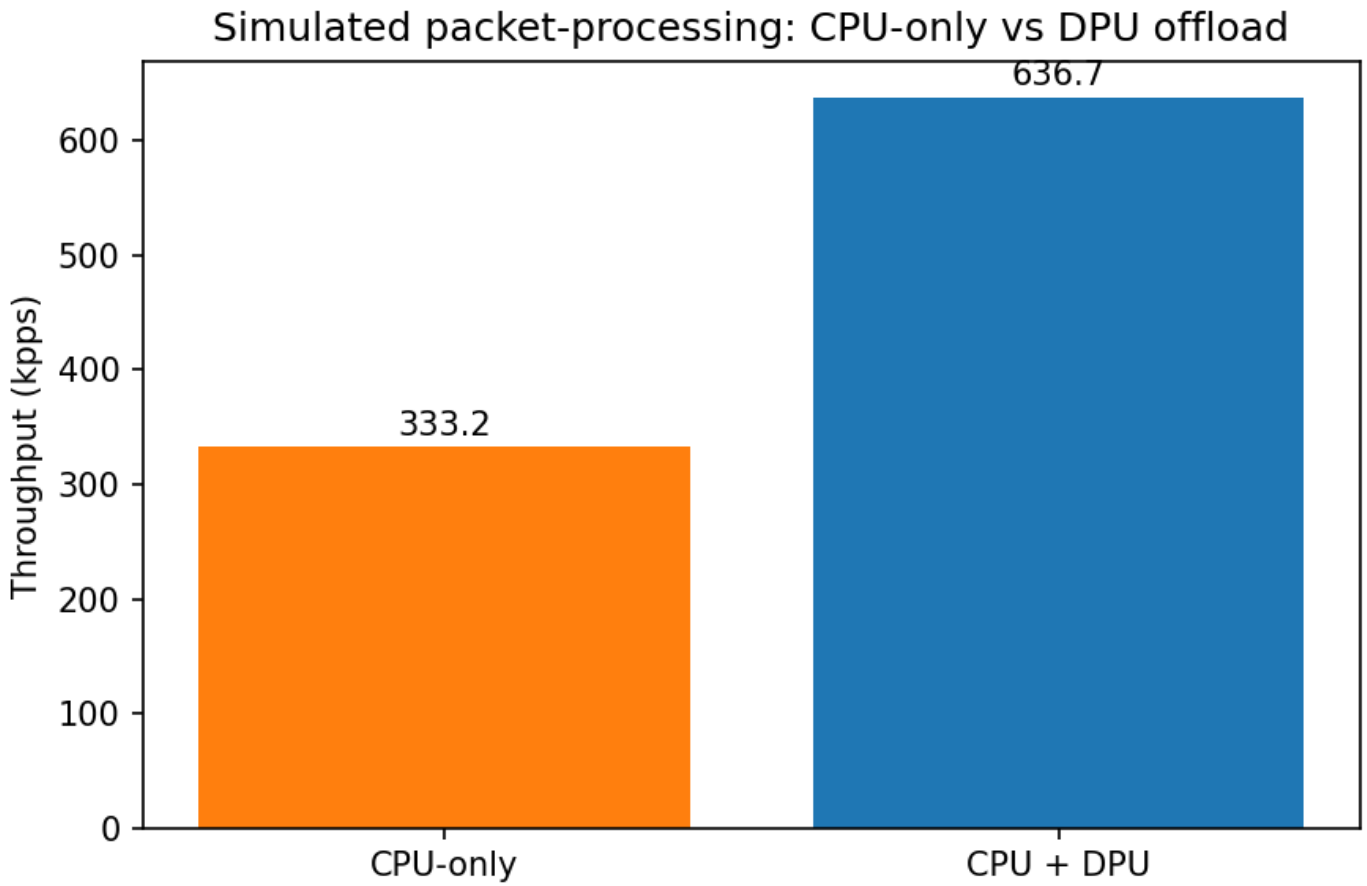
- Cloud providers (AWS, Azure) want to rent "Bare Metal" to users.
- **Problem**: How to monitor and isolate the user if they have full control?
- **Solution**: The DPU acts as the "Sidecar" manager. It manages networking and security *outside* the host, so the user cannot bypass it.

Application 3: Line-rate Security & DDoS Mitigation

- **Problem**: Volumetric attacks and high-rate malicious flows can saturate host CPUs and network queues.
- **DPU approach**: perform early-drop and per-flow state in the NIC (P4 / hardware match-action) or attach XDP/eBPF at the driver for rapid filtering.
- **Benefits**: drop malicious traffic at line-rate, reduce host CPU consumption, and enable real-time telemetry for upstream mitigation.
- **Demo hint**: start with ‘examples/03-ebpf-xdp‘ for early-drop logic; scale to DPU offloads for production.

Simulation: Offload Performance

We simulated the host CPU throughput gains when offloading work to a DPU.



Model note:

offloading fraction α of packet processing increases host capacity approximately proportionally.

Simulation results show $>$ 50% throughput increase in targeted workloads when logic is moved to the hardware pipeline.

Current Challenges and Limitations

- **Vendor Lock-in**: No universal "Standard" for all DPUs yet (though P4 helps).
- **Complex Debugging**: Inspecting code running inside a NIC is harder than on a host.
- **Power Consumption**: High-end DPUs can consume over 75W-100W per card.

Key Takeaways

- **Offload is Mandatory**: At 100G+ speeds, host processing is no longer viable.
- **Isolation is Security**: DPUs create a "Hard Gap" between the user VM and the infrastructure provider.
- **The New Tier**: DPUs are becoming the third pillar of data center compute alongside CPUs and GPUs.

Summary

- DPUs recover "lost" CPU cycles for user applications.
- They provide hardware-level security isolation.
- Programming models (P4/eBPF) are maturing fast.

Thank You!

Questions?

References

- Packet Pushers – DPU-based smart interfaces: `https://packetpushers.net/blog/dpu-based-smart-interfaces-and-th`
- Ubuntu – Data centre networking & SmartNICs: `https://ubuntu.com/blog/data-centre-networking-smartnics`
- Hot Chips – NVIDIA DPU (Idan Burstein): `https://www.hc33.hotchips.org/assets/program/conference/day1/HC2`
- Bosshart et al., "P4: programming protocol-independent packet processors," ACM SIGCOMM CCR, 2014.
- Hoiland-Jorgensen et al., "The eXpress Data Path (XDP)," CoNEXT 2018.
- G. Liu et al., "A Survey of SmartNICs and DPUs," IEEE Communications Surveys & Tutorials, 2024.
- P4 Language Consortium – `https://p4.org`
- eBPF resources – `https://ebpf.io`