Mélyen programozható hálózatok, avagy SDN 2.0

Dr. Laki Sándor, adjunktus

ELTE Communication Networks Laboratory

HUNOG_2, 2024.10.09-10, Inárcs

Paradigma váltás a hálózatokban

- Zárt HW és SW rendszerek
- Több 100 protokollt implementálnak
- Magas fogyasztás

Nehezen menedzselhető
Megbízhatatlan
Nehezen skálázható, bővíthető
Lassú innováció

- Whitebox switch-ek
  + control plane-t futtató szerverek
- Control plane – Data plane szeparáció
- Merchant silicon
- Linux alapú
- Open Source Swich OS és software

Könnyebb hibajavítás
Új lehetőségek
- Traffic engineering
- Meghibásodás kezelése
- Biztonságosság

Gyorsabb innováció

Paradigma váltás a hálózatokban

- Zárt HW és SW rendszerek
- Több 100 protokollt implementálnak
- Magas fogyasztás

Nehezen menedzselhető
Megbízhatatlan
Nehezen skálázható, bővíthető
Lassú innováció

- Whitebox switch-ek
  + control plane-t futtató szerverek
- Control plane – Data plane szeparáció
- Merchant silicon
- Linux alapú
- Open Source Swich OS és software

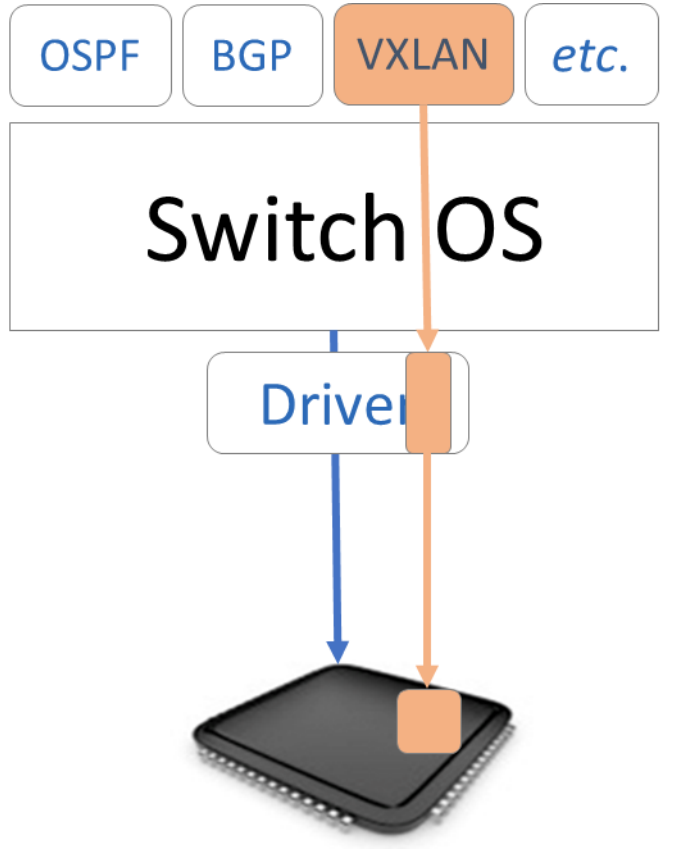
SDN 1.0

Könnyebb hibajavítás
Új lehetőségek
- Traffic engineering
- Meghibásodás kezelése
- Biztonságosság

Gyorsabb innováció

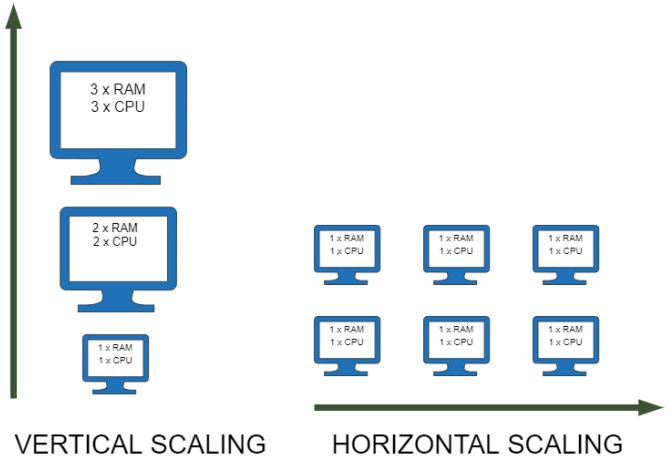
SDN mindent megold…

- Control plane programozható
- A legtöbb protokoll egyedi csomagfeldolgozást is igényel
- Igény a data plane programozhatóságára a csomagfeldolgozás testreszabására



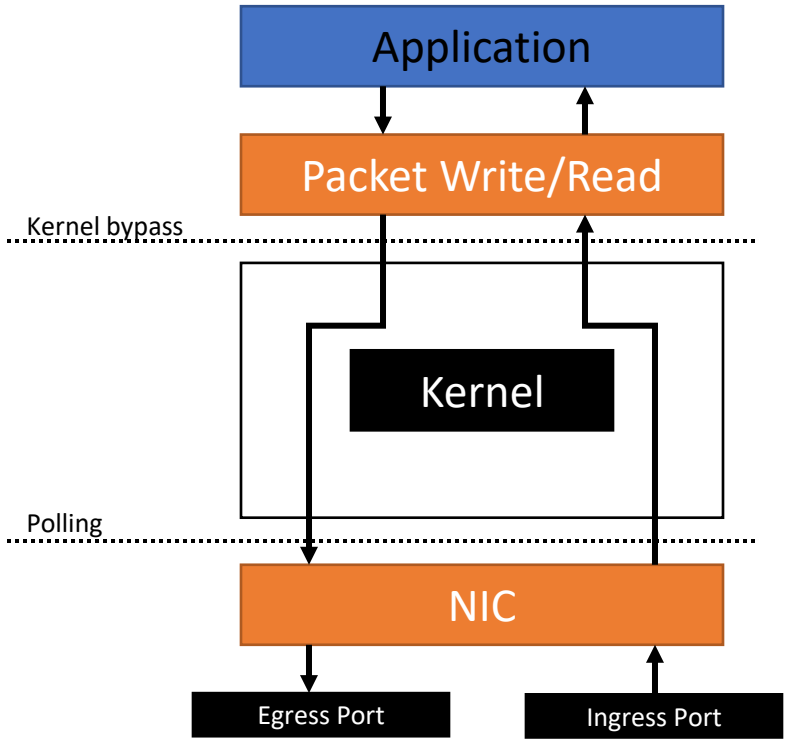
Paradigma váltás a hálózatokban

- Új funkciók bevezetése
  - Gyorsan és testre szabhatóan
- Szoftverizált csomagfeldolgozás
  - Csomagok kezelése szoftverben
  - Hagyományos szervereken
- Rugalmasság és skálázhatóság
  - Szofver példányok fel- és leskálázása
  - Network Function Virtualization



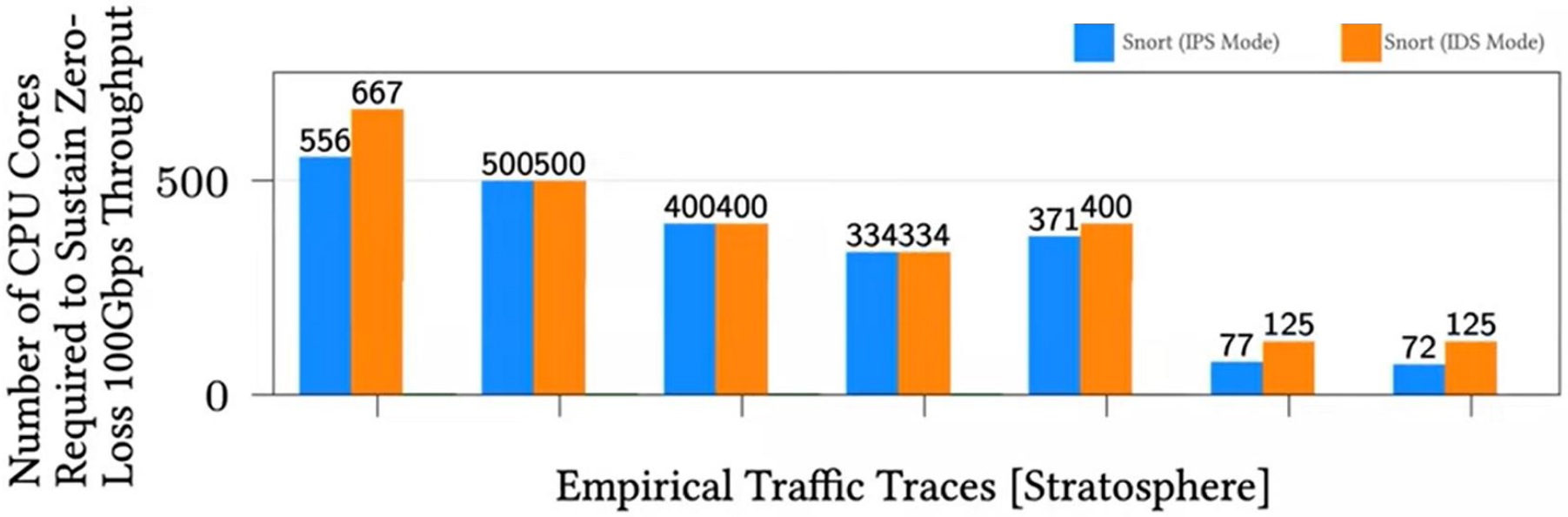
Problémák…

- Megbecsülhetetlen késleltetés, nem garantálható alacsony késleltetés
  - x86 architektúra nem csomagfeldolgozási feladatra lett tervezve
- Sávszélesség korlátok
  - Számos szűk keresztmetszet: PCIe sebesség, cache problémák, memória hozzáférés etc.
- Kernel-bypass technikák
  - Nagy teljesítmény CPU-n
  - Megfelelő sávszélsesség
  - 100% CPU kihasználtság
  - Folyamatos NIC polling
- Energia fogyasztás nagy
  - Magas W/pps
  - Üzemelési költség OPEX



Egy példa 1.0 - SNORT

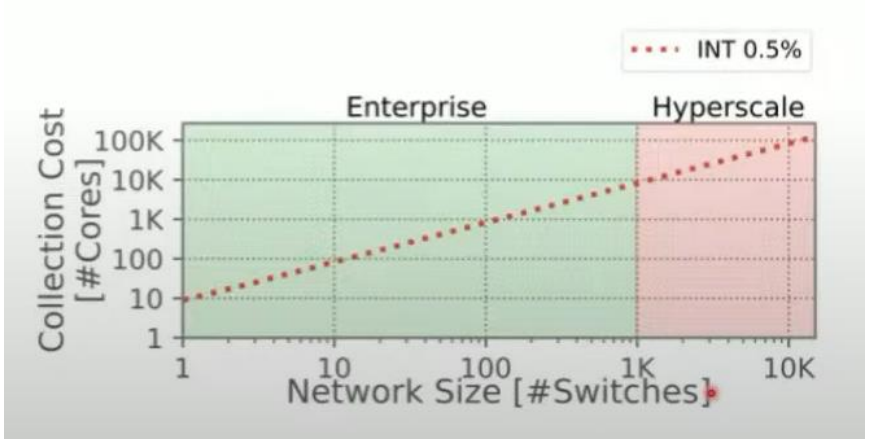
- Snort 3.0 alkalmazási szintű tűzfal
- 100 Gbps, csomagvesztés nélkül

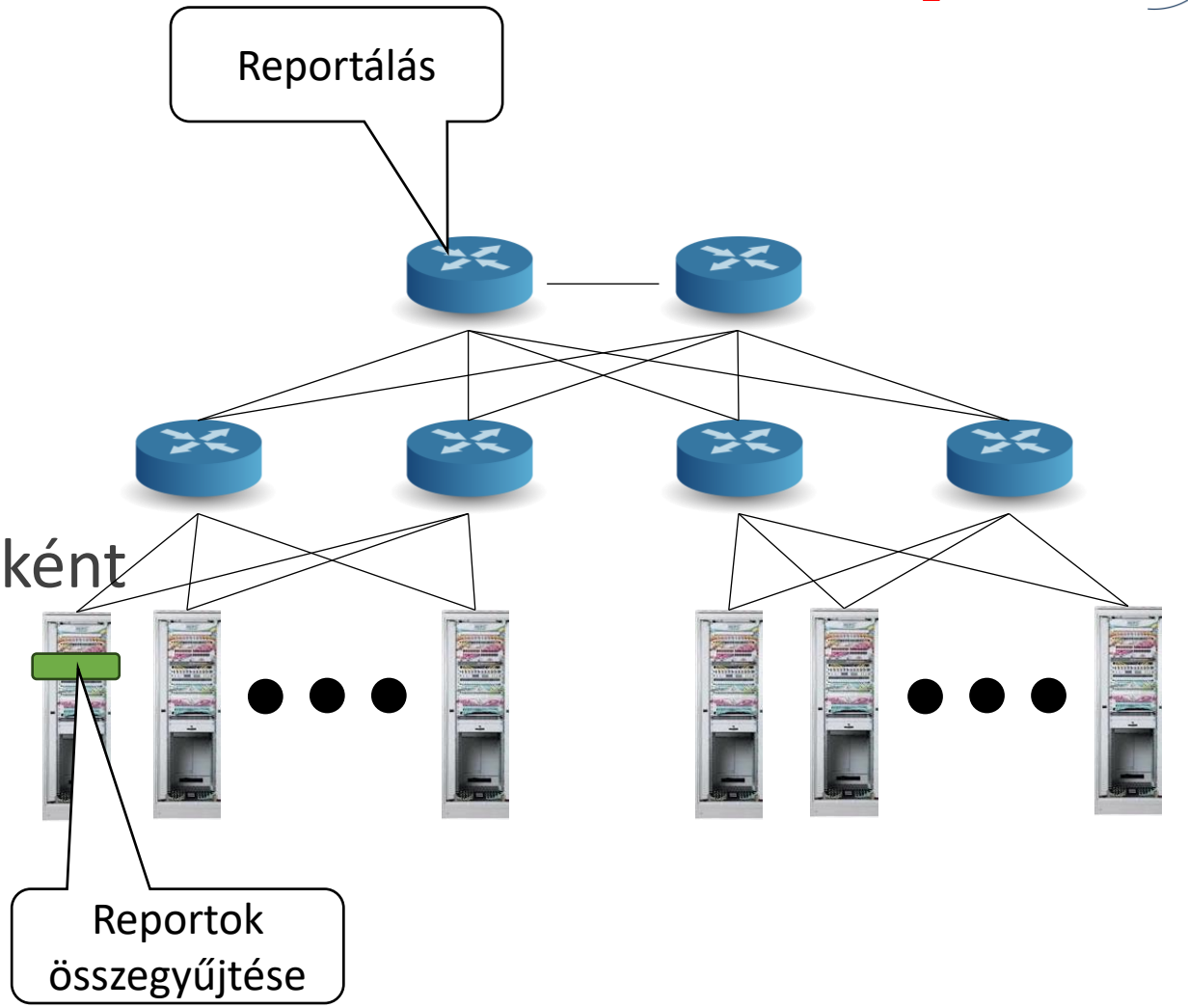


Source: Justine Sherry: Re-envisioning generic server architectures for I/O driven compute [EuroP4]

Példa 2.0 – Telemetry collection

- Hálózat telemetria adatok gyűjtése
- Hyperscale data center
  - Több ezer switch
- Néhány millió report másodpercenként switchenként
  - Alibaba - ACM SIGCOMM 2020





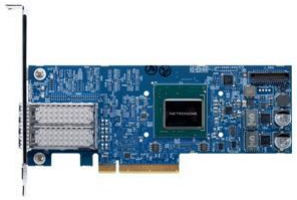
Source: Gianni Antichi: To offload or not to offload

Költséges felskálázás

- Eszköz/Hardver költség
- Áram költség
- Rackspace költség

Programozható hálózati eszközök

- SmartNIC



- Programozható Switch

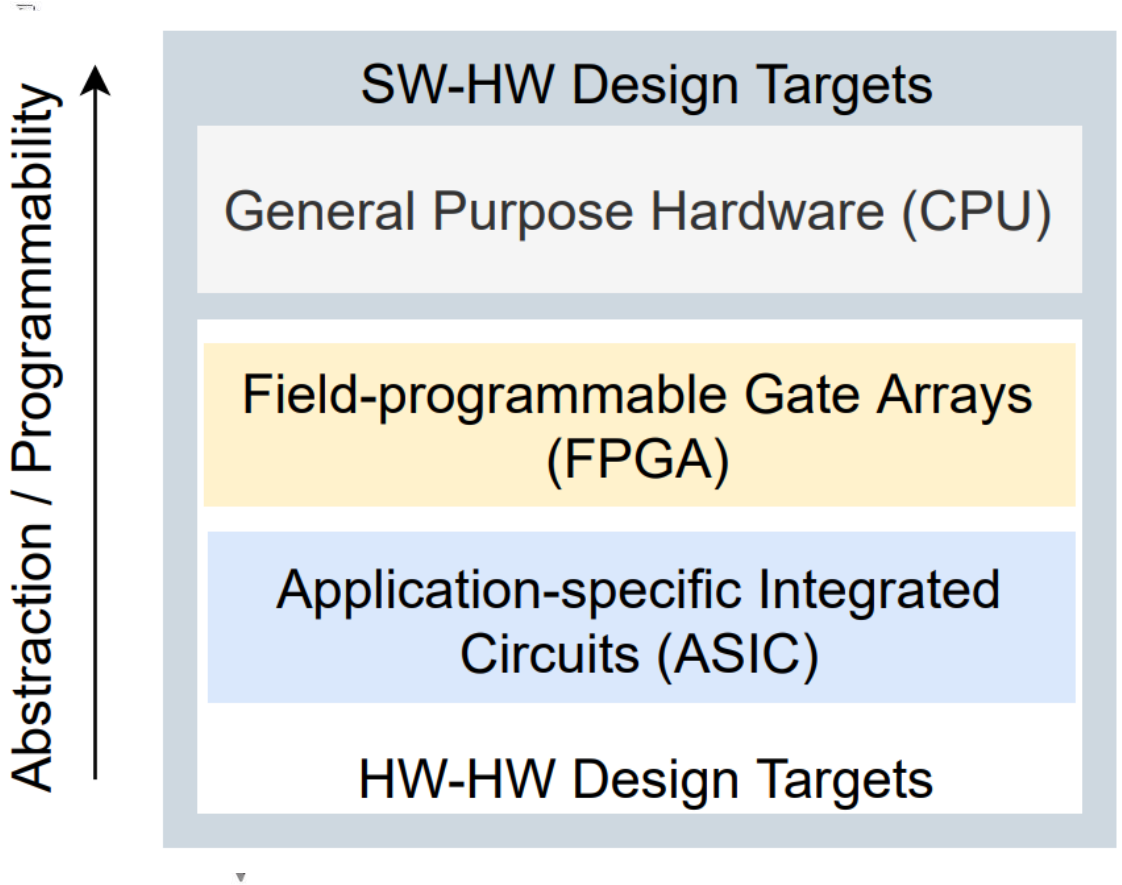
- DPU/IPU

- FPGA alapú NIC

- Programozási nyelvek
  - P4, NPL
- Könyvtárak
  - DOCA, eBPF/XDP, DPDK Flow API
- FPGA fejlesztő környezetek

Programozható hálózati eszközök

- SmartNIC
- Programozha
- DPU/IPU
- FPGA alapú I



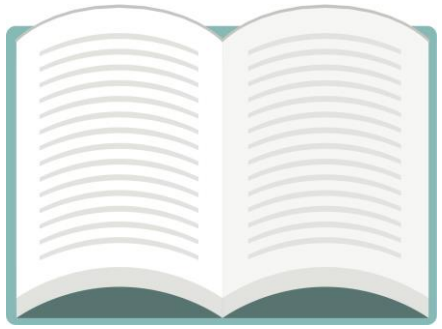
ek

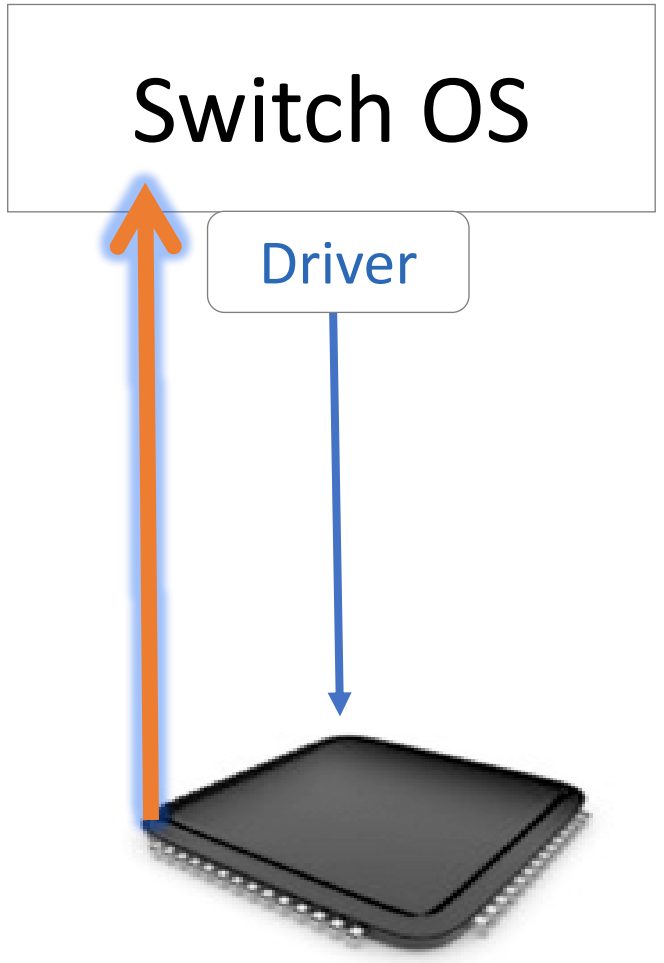
)PDK Flow API

yezetek

SDN 1.0

“This is how I process packets …”





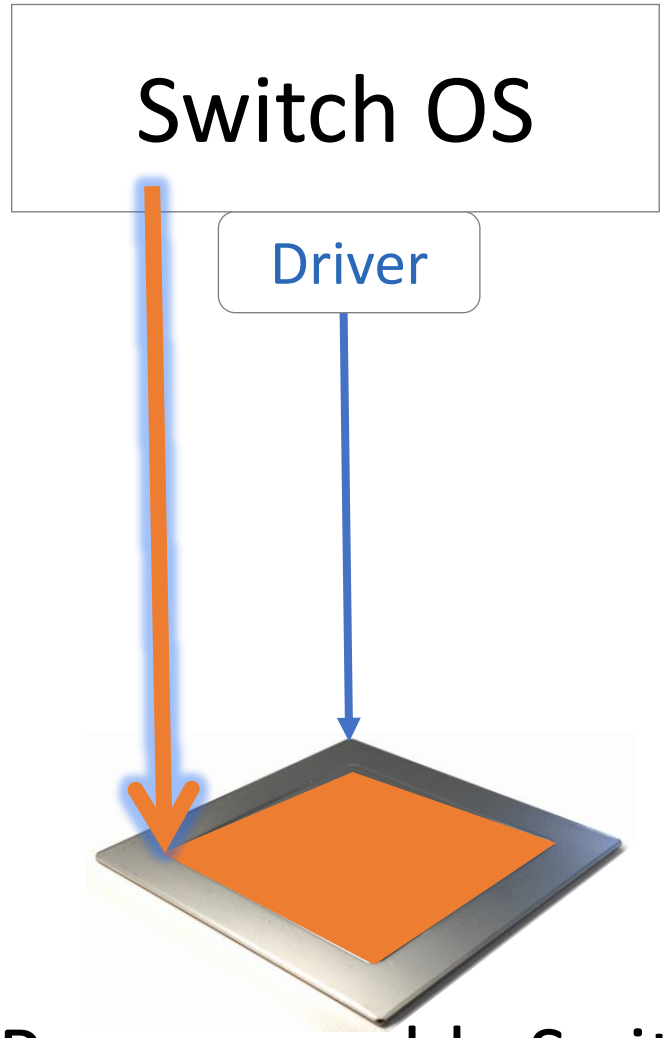
Fixed-function switch

Mély programozhatóság SDN 2.0

“This is precisely how you must process packets”

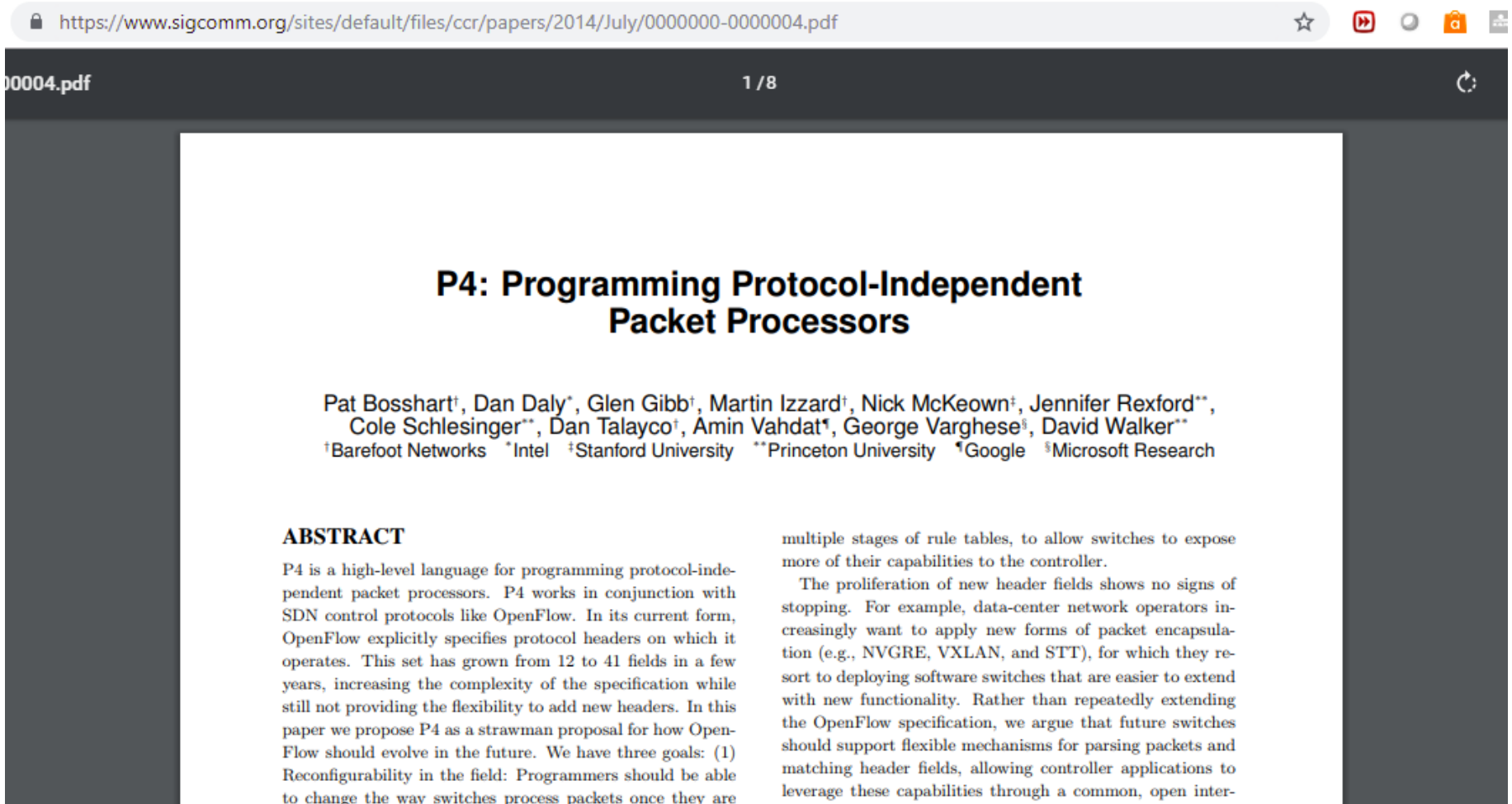
```
table int_table {
  reads {
    ip.protocol;
  }
  actions {
    export_queue_latency;
  }
}
```

```
action export_queue_latency (sw_id) {
  add_header(int_header);
  modify_field(int_header.kind, TCP_OPTION_INT);
  modify_field(int_header.len, TCP_OPTION_INT_LEN);
  modify_field(int_header.sw_id, sw_id);
  modify_field(int_header.q_latency,
               intrinsic_metadata.deq_timedelta);
  add_to_field(tcp.dataOffset, 2);
  add_to_field(ipv4.totalLen, 8);
  subtract_from_field(ingress_metadata.tcpLength,
                      12);
}
```



P4.org

https://www.sigcomm.org/sites/default/files/ccr/papers/2014/July/0000000-0000004.pdf



https://www.sigcomm.org/sites/default/files/ccr/papers/2014/July/0000000-0000004.pdf

00004.pdf 1 / 8

P4: Programming Protocol-Independent Packet Processors

Pat Bosshart†, Dan Daly*, Glen Gibb†, Martin Izzard†, Nick McKeown‡, Jennifer Rexford**, Cole Schlesinger**, Dan Talayco†, Amin Vahdat¶, George Varghese§, David Walker**
†Barefoot Networks *Intel ‡Stanford University **Princeton University ¶Google §Microsoft Research

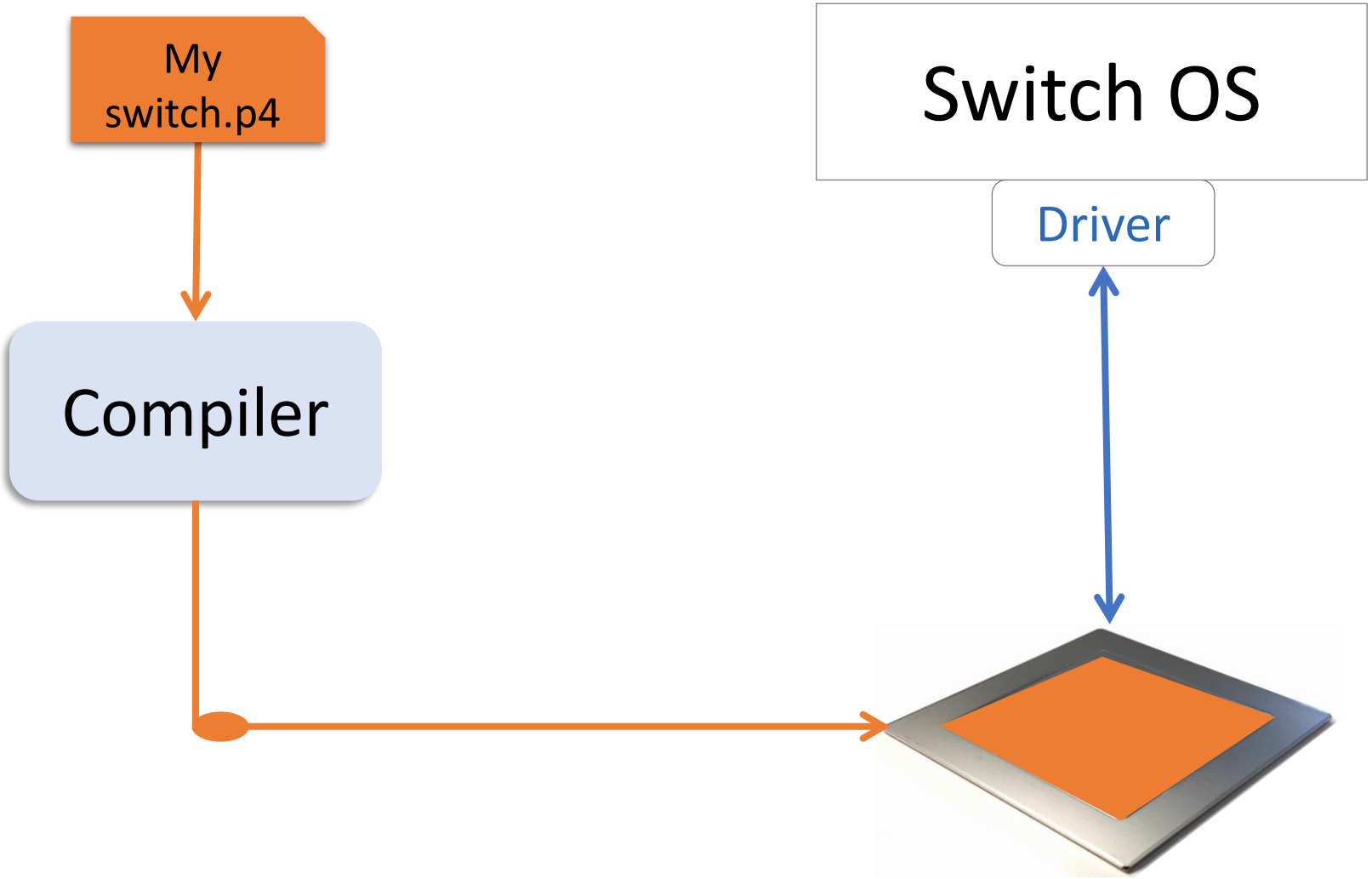
ABSTRACT

P4 is a high-level language for programming protocol-independent packet processors. P4 works in conjunction with SDN control protocols like OpenFlow. In its current form, OpenFlow explicitly specifies protocol headers on which it operates. This set has grown from 12 to 41 fields in a few years, increasing the complexity of the specification while still not providing the flexibility to add new headers. In this paper we propose P4 as a strawman proposal for how OpenFlow should evolve in the future. We have three goals: (1) Reconfigurability in the field: Programmers should be able to change the way switches process packets once they are

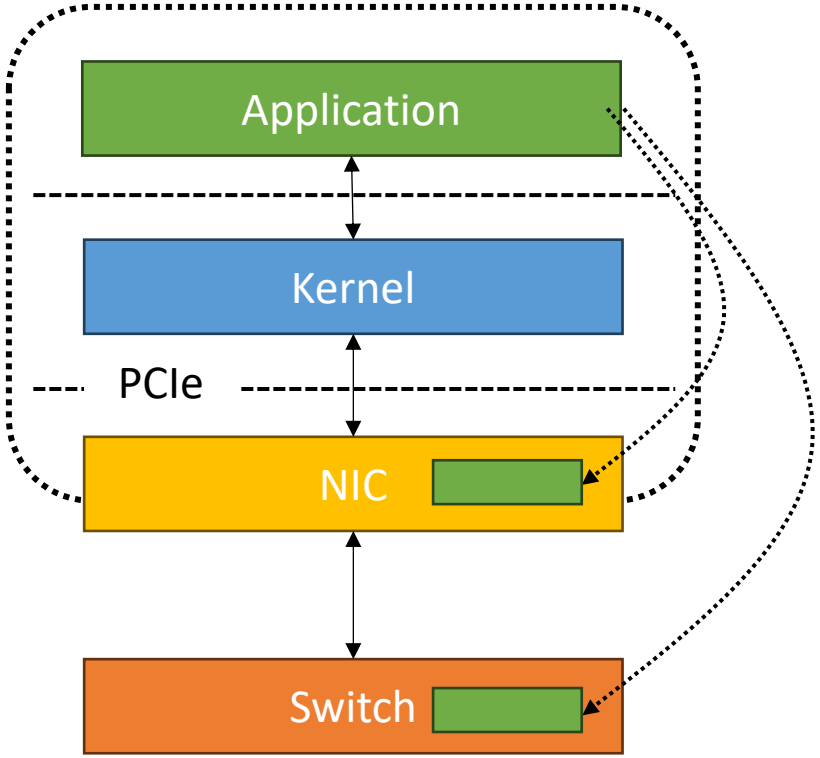
multiple stages of rule tables, to allow switches to expose more of their capabilities to the controller.

The proliferation of new header fields shows no signs of stopping. For example, data-center network operators increasingly want to apply new forms of packet encapsulation (e.g., NVGRE, VXLAN, and STT), for which they resort to deploying software switches that are easier to extend with new functionality. Rather than repeatedly extending the OpenFlow specification, we argue that future switches should support flexible mechanisms for parsing packets and matching header fields, allowing controller applications to leverage these capabilities through a common, open inter-

Reducing complexity



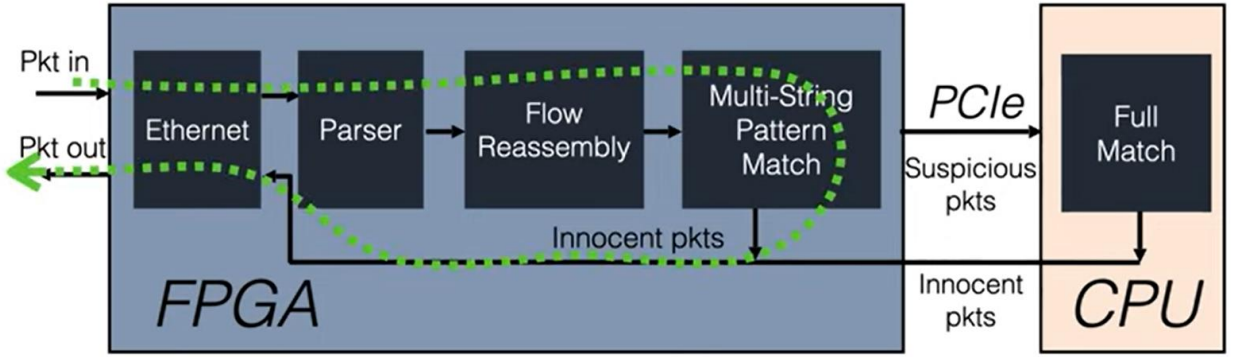
Programozható csomagfeldolgozási lehetőségek és offloadolás

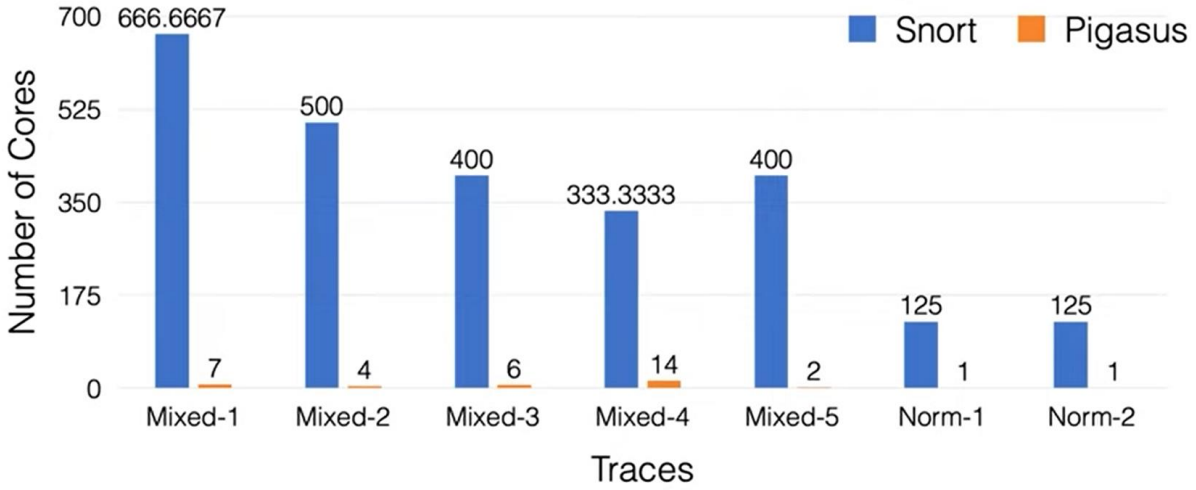


- Hardware offloading
  - Csomag feldolgozási logika kiszervezése a NIC-re vagy a switchekre
  - Kevesebb PCIe tranzakció
  - Kevesebb allokált skb a kernelben
  - Kevesebb I/O művelet
  - CPU ciklus megtakarítás

Példa 1.0 - SNORT

- Multi-String Pattern Matching
- Hibrid architektúra
- FPGA alapú NIC, mint gyorsító
  - Egyszerűbb szabályok
- A terhelés 95%-án a NIC dönt
- 5% esetében a CPU bevonása is szükséges

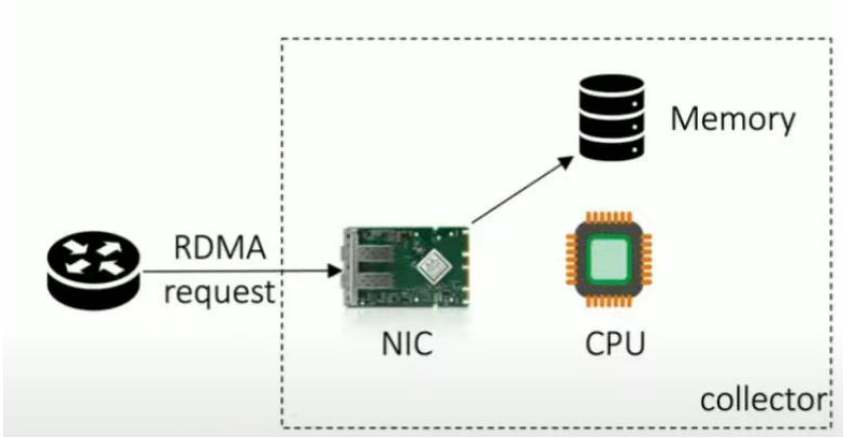


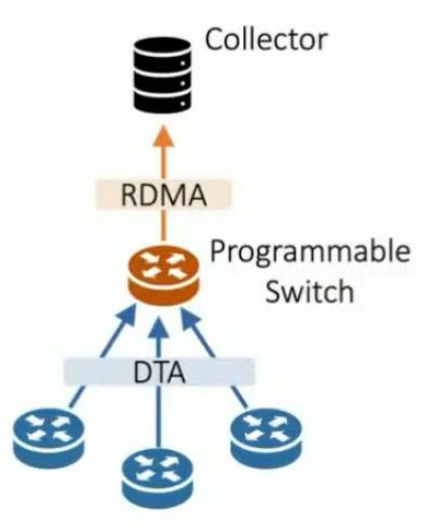


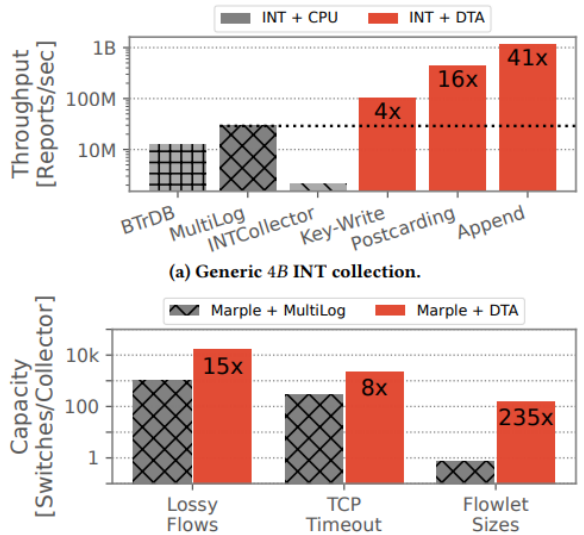
Source: Justine Sherry: Re-envisioning generic server architectures for I/O driven compute [EuroP4]

Példa 2.0 – Telemetry Collector

- RDMA közvelten memória hozzáférés a CPU kihagyásával
  - Közvetlen címzés kell
  - Data plane számára bonyolult
- DTA egyszerű protokoll
  - Kicsi overhead és komplexitás
- Collector switch
  - DTA -> RDMA fordítást végez
  - Címzés kezelése, stb.
- Switchek száma/collector javulás



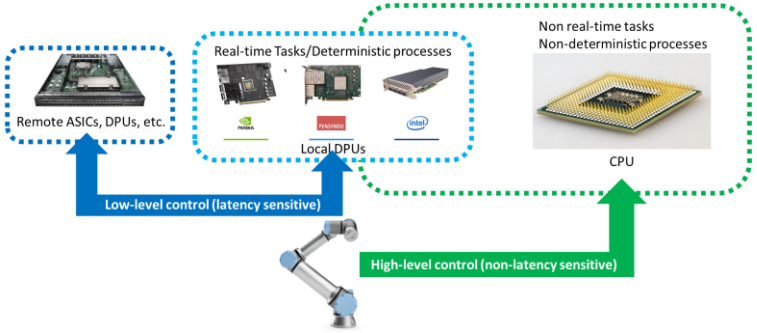




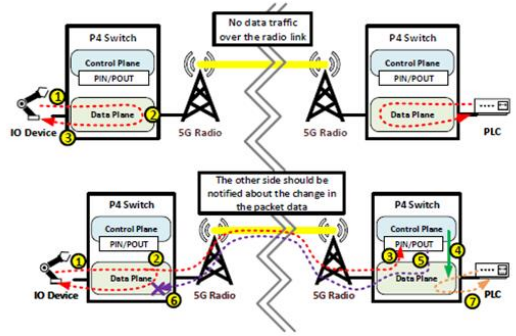
Source: J. Langlet et al.: Direct Telemetry Access, ACM SIGCOMM 2023

További példák

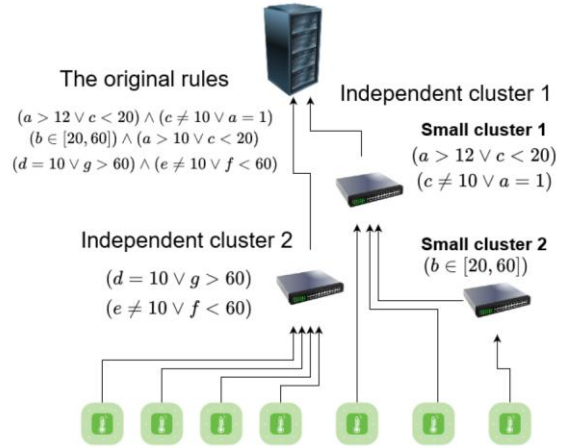
- Towards real-time cloud with in-network computing
  - Offloading low-level velocity control of robot arms
  - Supporting **deterministic EDGE-Computing** applications
  - ROS-integration
  - S. Laki et al., USENIX NSDI 2022



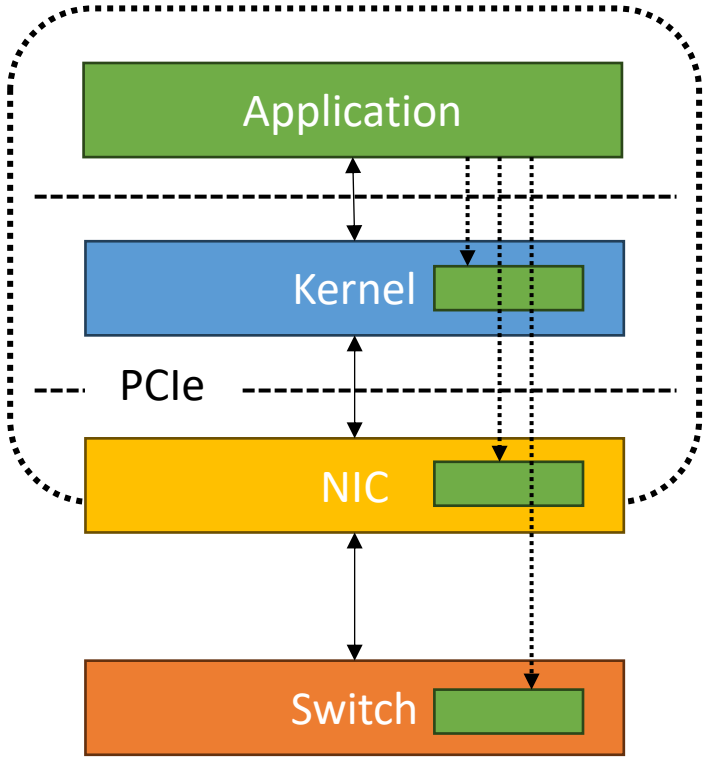
- Adaptive traffic reduction with in-network caching and filtering
  - ProfiNet traffic is redundant
  - Traffic reduction on a critical link
  - Cs. Györgyi et al., IEEE Access 2023



- Distributed in-network event detection in sensor streams
  - Disaggregated pipeline for event detection in the network
  - Better scalability than with a single device
  - Faster reaction than with the server-based counterparts like Apache Flink
  - Cs. Györgyi et al., IEEE NetSoft 2022



Konklúzió



- Alkalmazások teljesítményének felskálázása
  - Nem csak hagyományos hálózati feladatok
- Hol tartunk
  - Teljes programozható stack
    - Kernel és hardver
    - Nyelvek és könyvtárak – P4, eBPF/XDP, DPDK
  - Új eszközök
    - Teljesítény + Programozhatóság
- Nehézségek
  - Technológiai érettség?
  - Melyik funkciót érdemes kiszervezni?
  - Fejlesztői képességek

ELTE
EÖTVÖS LORÁND
UNIVERSITY

http://lakis.web.elte.hu