Future of Programmable Packet Processing

Our part in making networks better

Changhoon Kim
Intel Fellow
CTO of Applications, Barefoot Networks Division, Intel

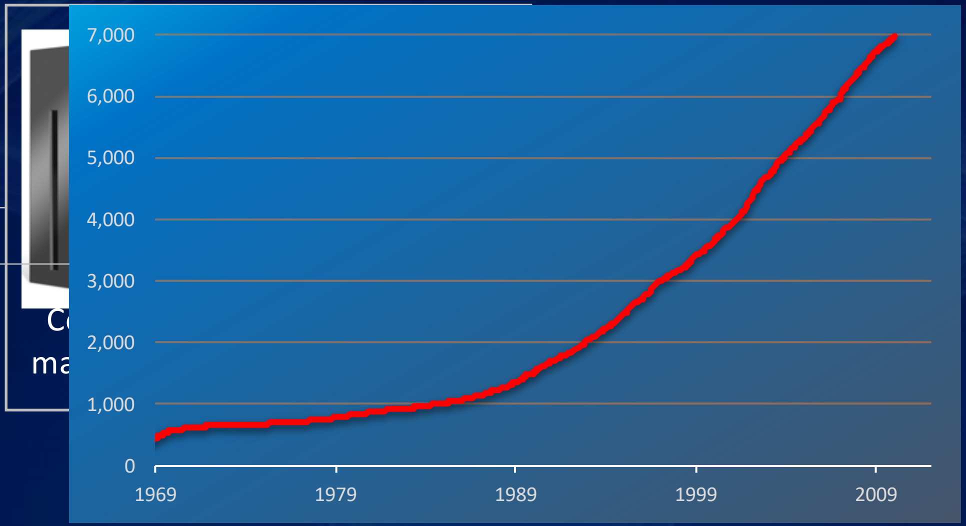
2020

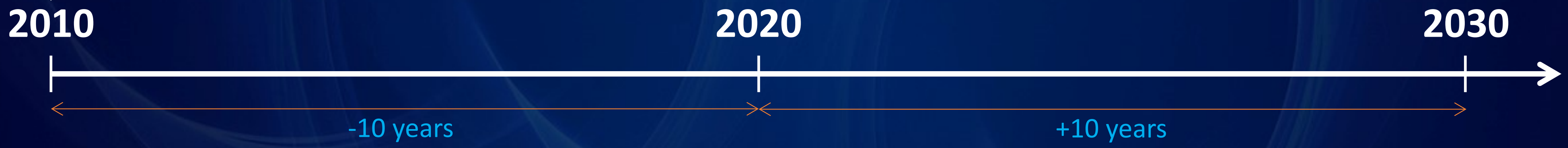
2030

+10 years

“closed and proprietary”
“proliferation of standards”
“barrier to entry”
“stranglehold by vendors”

Number of IETF RFCs




2010

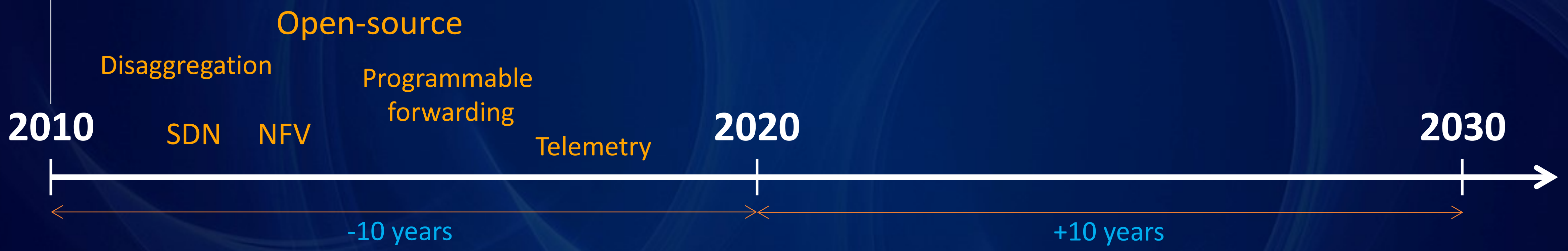
2020

2030

-10 years

+10 years

“closed and proprietary”
“proliferation of standards”
“barrier to entry”
“stranglehold by vendors”



Open-source

Disaggregation

Programmable forwarding

2010

SDN

NFV

Telemetry

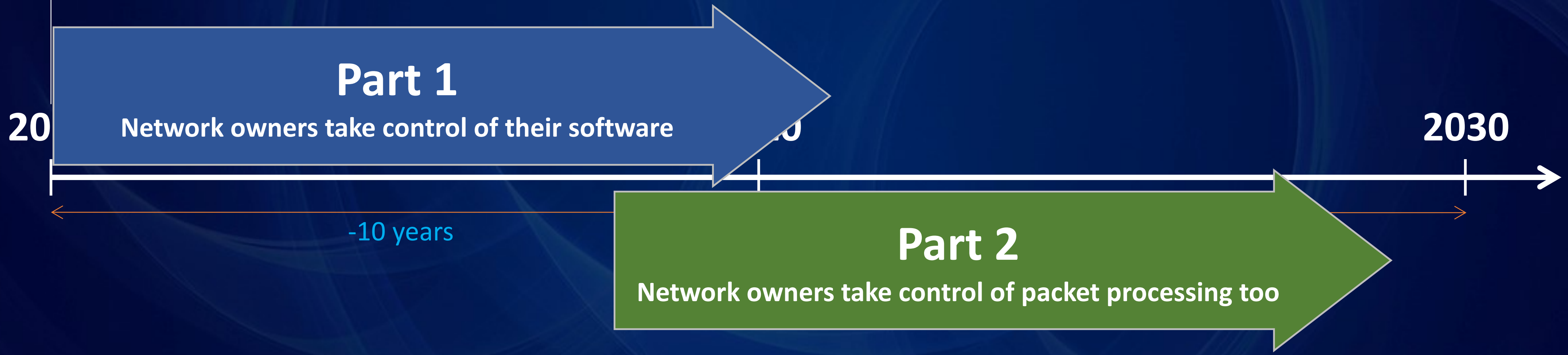
2020

2030

-10 years

+10 years

“closed and proprietary”
“proliferation of standards”
“barrier to entry”
“stranglehold by vendors”



20

2030

Part 1

Network owners take control of their software

-10 years

Part 2

Network owners take control of packet processing too

We devoted 6 years to programmable packet processing

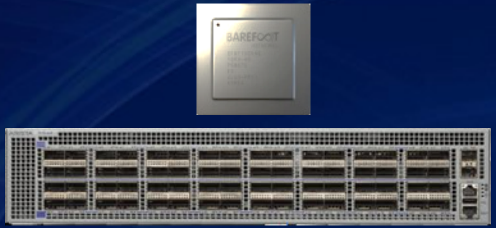
Now we observe:

Programmable switch chips can have the same power, performance and cost as fixed function switches.

Beautiful new ideas are now owned by the programmer, not the chip designer.

Which means more innovation.

But, how do we know if a programmable switch chip has the same power, performance and cost as a fixed-function switch chip?



Feature	P4 Programmable “Tofino”	Fixed Function ASIC	Benefits of P4-programmability
L2/L3 Throughput	6.4Tb/s	6.4Tb/s	
Number of 100G Ports	64	64	
Availability	Yes	Yes	
Max Forwarding Rate	4.8B packets per sec	4.2B packets per sec	14% Greater Performance
Max 25G/10G Ports	256/258	128/130	
Programmability	Yes (P4)	No	
Typical System Power draw	4.2W per port	4.9W per port	14% Lower Power
Large Scale NAT	Yes (100k)	No	
Large scale stateful ACL	Yes (100k)	No	
Large Scale Tunnels	Yes (192k)	No	
Packet Buffer	Unified	Segmented	Better Burst Absorption
Segment Rtg/Bare Metal	Yes/Yes	No/No	
LAG/ECMP Hash Algorithm	Full entropy, programmable	Hash seed, reduced entropy	
ECMP	256 way	128 way	
Telemetry and Analytics	Line-rate per flow stats	Sflow (Sampled)	Real-time Visibility

Advances in programmable packet processing in the last few years across the industry

- High-speed switching ASICs – at least three major vendors
- Several types of smart NICs – every major vendor, a few MSDCs, and several new start-ups
- High-speed S/W-based packet processing – OVS, DPDK, eBPF, VPP, etc.
- Real-world deployment at scale – by MSDCs and carriers for various roles
- Packet-level data-plane telemetry (INT) – one of the hottest technologies developed and deployed today
- Early attempts of real-time closed-loop control put in practice

Programmable packet processing is becoming the mainstream

A Vision for 2030

1. NICs, Switches, vSwitches, end host networking stacks will have been programmable for >7 years.

2. We will think of a network as a programmable platform. Behavior described at top; partitioned, compiled and run across elements.

3. Every DC will work differently, programmed and tailored locally.

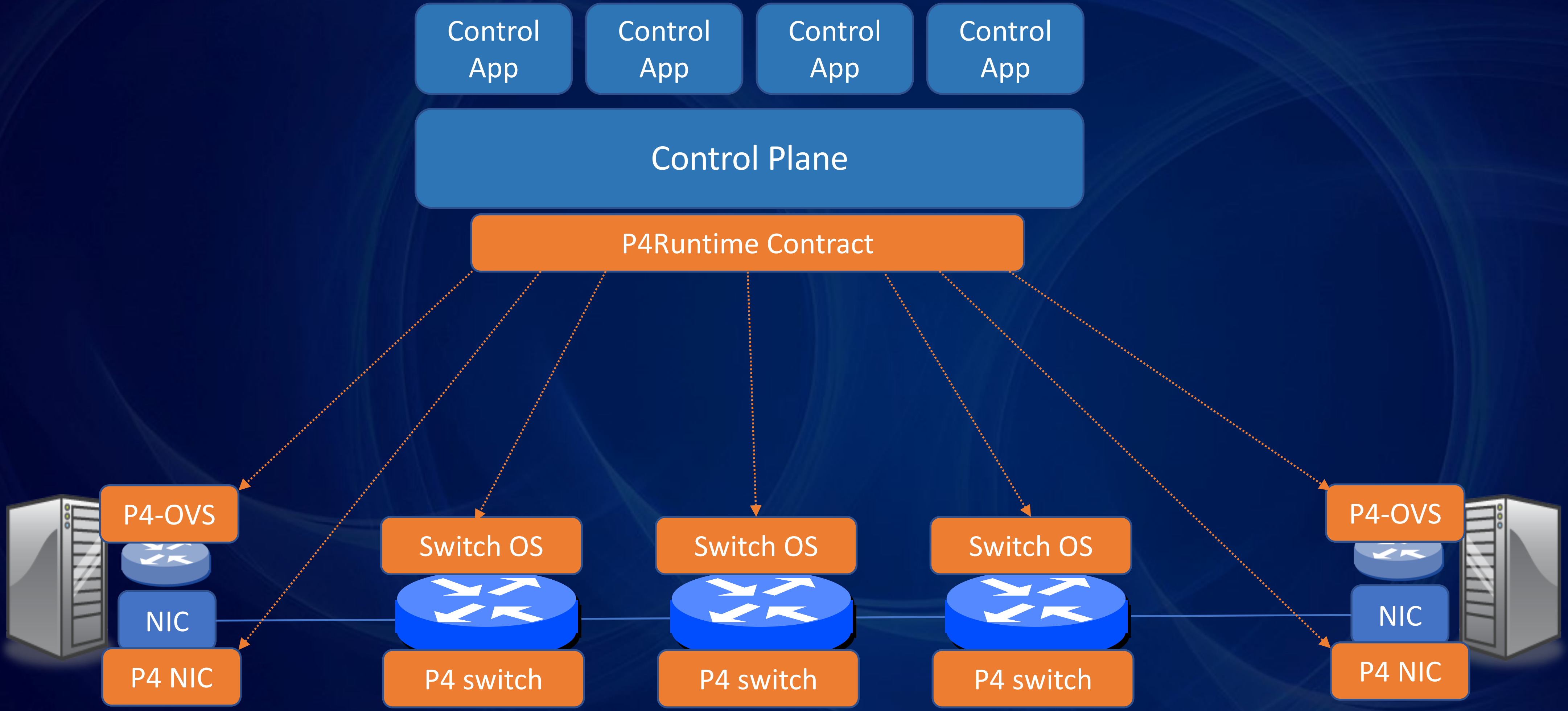
   e.g., Routing: Packets might be source-routed by topology-aware end-hosts.

   e.g., Congestion control: Might use direct knowledge of precise queue occupancy, not heuristics based on loss and RTT.

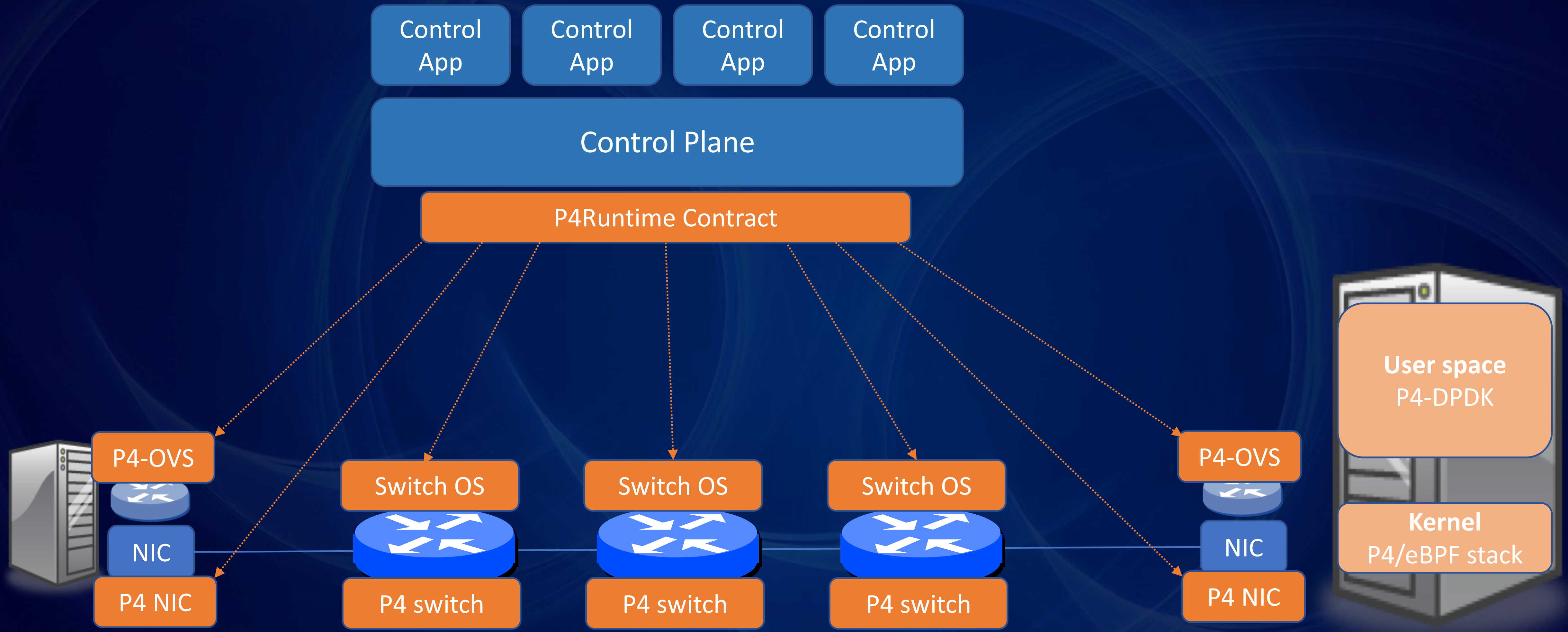
A vision of the network as a programmable platform

Network owners and operators will use **fine-grain measurement** and **PL/ML technologies** to automate network control at scale.

Example: Large Cloud or ISP



Example: Large Cloud or ISP



Control App

Control App

Control App

Control App

Control Plane

P4Runtime Contract

P4-OVS

NIC

P4 NIC

Switch OS

P4 switch

Switch OS

P4 switch

Switch OS

P4 switch

P4-OVS

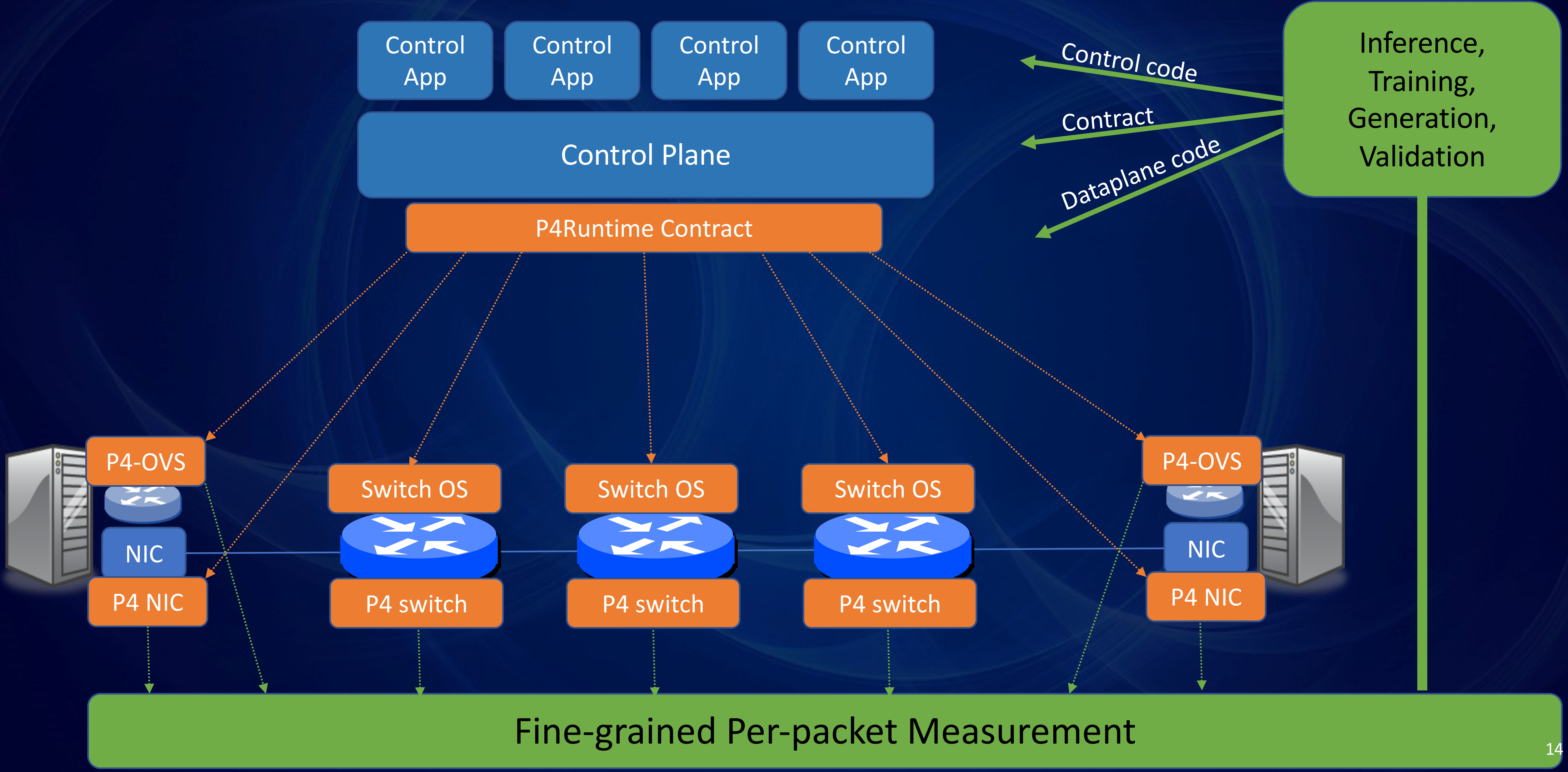
NIC

P4 NIC

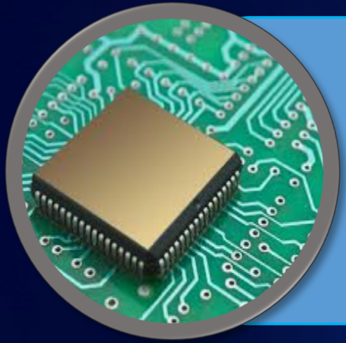
User space
P4-DPDK

Kernel
P4/eBPF stack

Example: Large Cloud or ISP



Our part to play at Intel



More Targets

- P4 switching ASICs, NICs and SmartNICs, FPGAs, and packet-processing S/W
- Hybrid targets
- Target P4 architectures

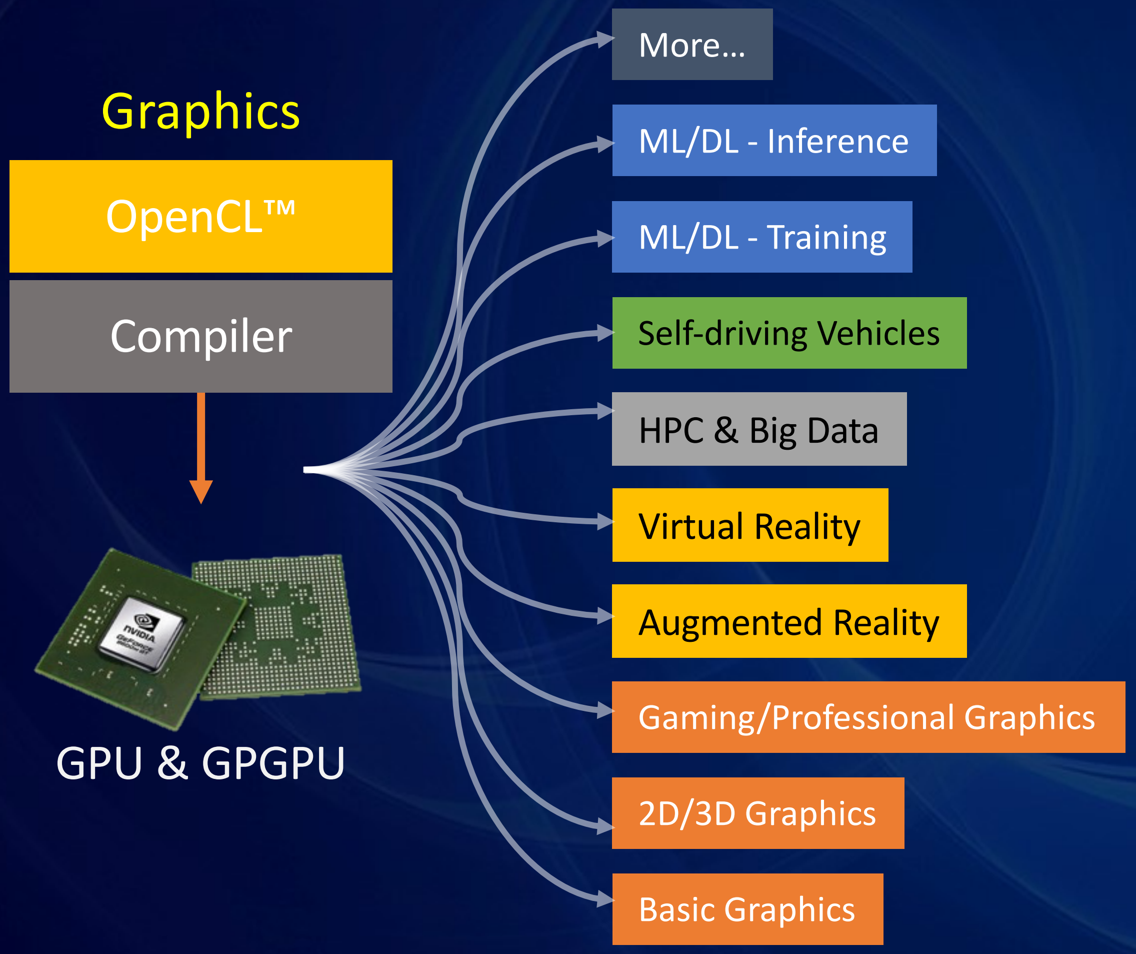
More Software

- Compilation, verification, validation, debugging, and test tools
- Run-time APIs and libraries
- A much bigger networking open-source community for development & education

More Apps

- End-to-end data-plane telemetry
- End-host / NIC applications (e.g., virtual switching, congestion control, message processing)
- Compute offloading (e.g., stream processing)

Cambrian explosion of beautiful new apps



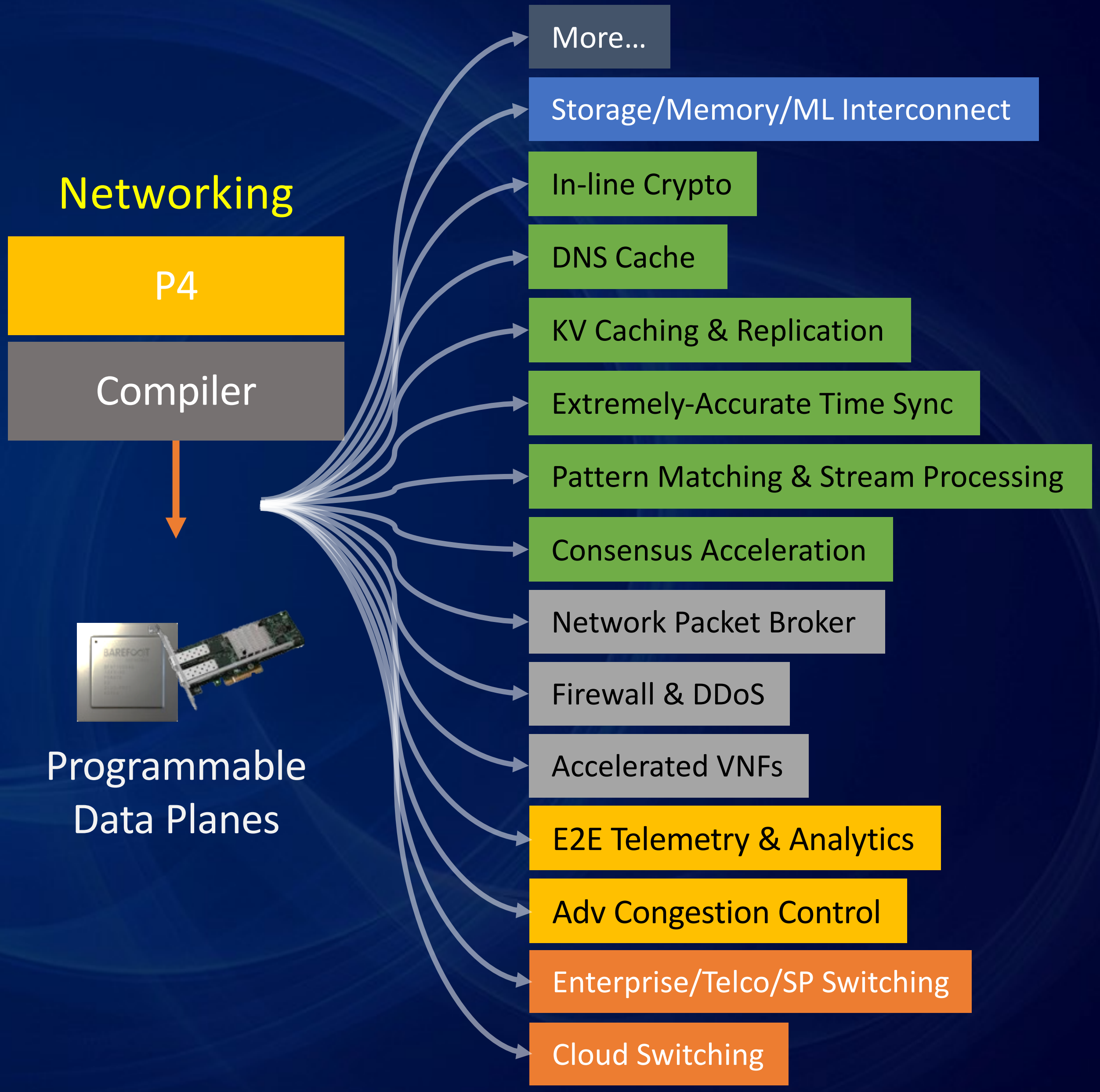
Graphics

OpenCL™

Compiler

GPU & GPGPU

- More…
- ML/DL - Inference
- ML/DL - Training
- Self-driving Vehicles
- HPC & Big Data
- Virtual Reality
- Augmented Reality
- Gaming/Professional Graphics
- 2D/3D Graphics
- Basic Graphics



Networking

P4

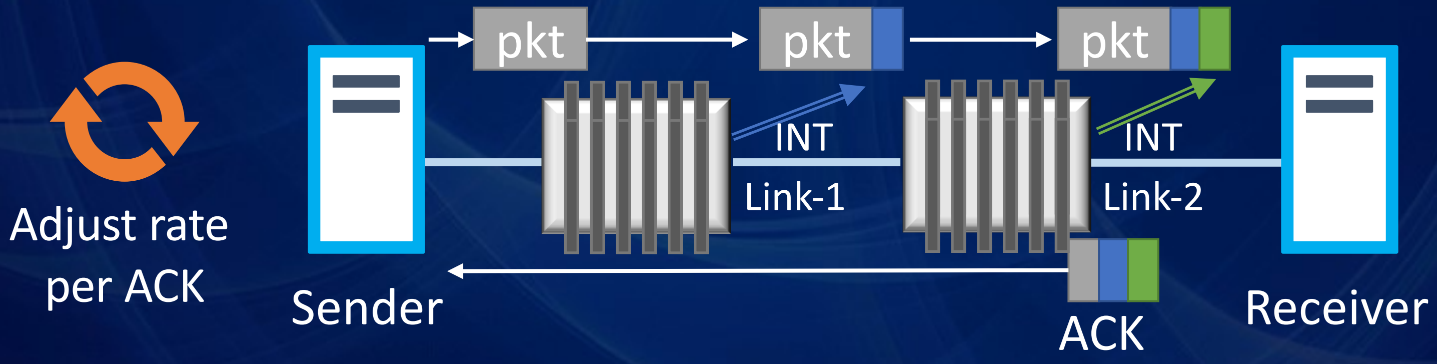
Compiler

Programmable
Data Planes

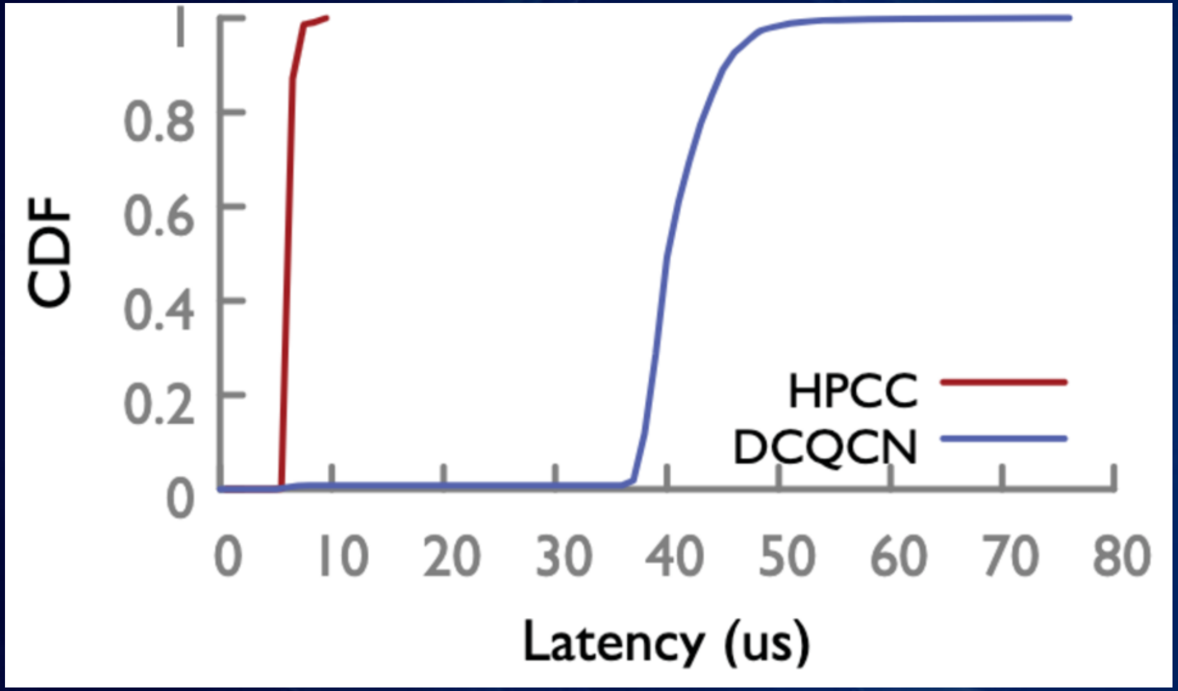
- More…
- Storage/Memory/ML Interconnect
- In-line Crypto
- DNS Cache
- KV Caching & Replication
- Extremely-Accurate Time Sync
- Pattern Matching & Stream Processing
- Consensus Acceleration
- Network Packet Broker
- Firewall & DDoS
- Accelerated VNFs
- E2E Telemetry & Analytics
- Adv Congestion Control
- Enterprise/Telco/SP Switching
- Cloud Switching

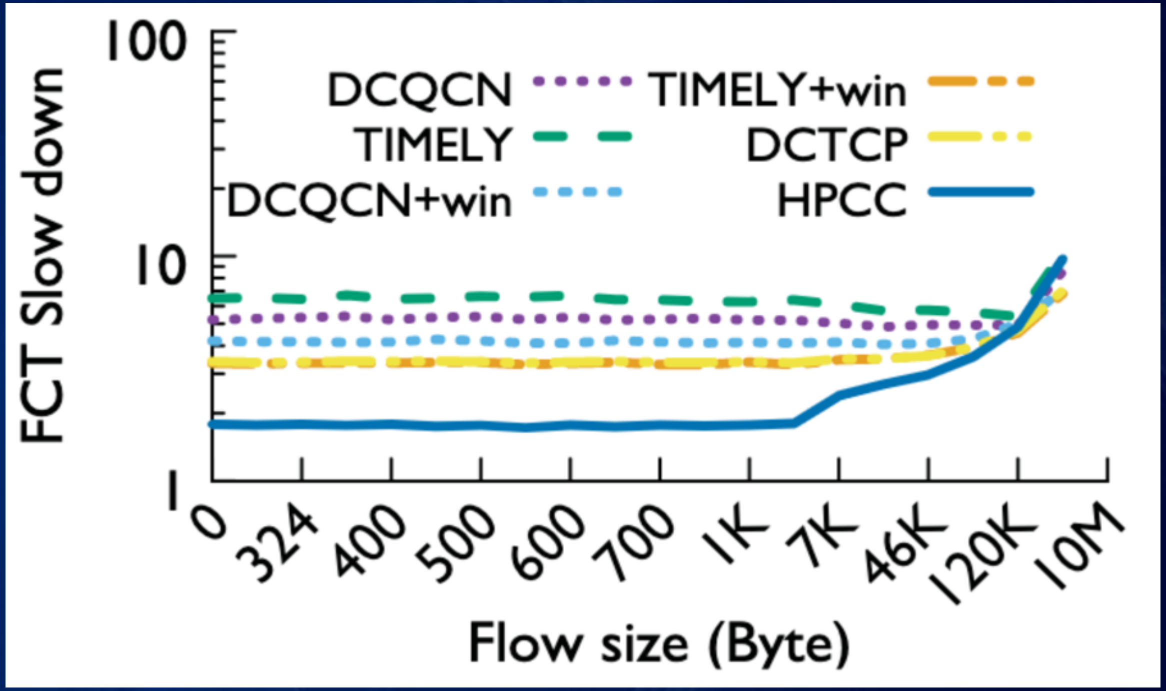
HPCC: INT-based High-Precision Congestion Control

- Developed and deployed by Alibaba and their collaborators
- Using INT as explicit and precise feedback
    - Very fast convergence via MIMD (multiplicative increase & multiplicative decrease)
    - Near-zero queue
    - Few parameters



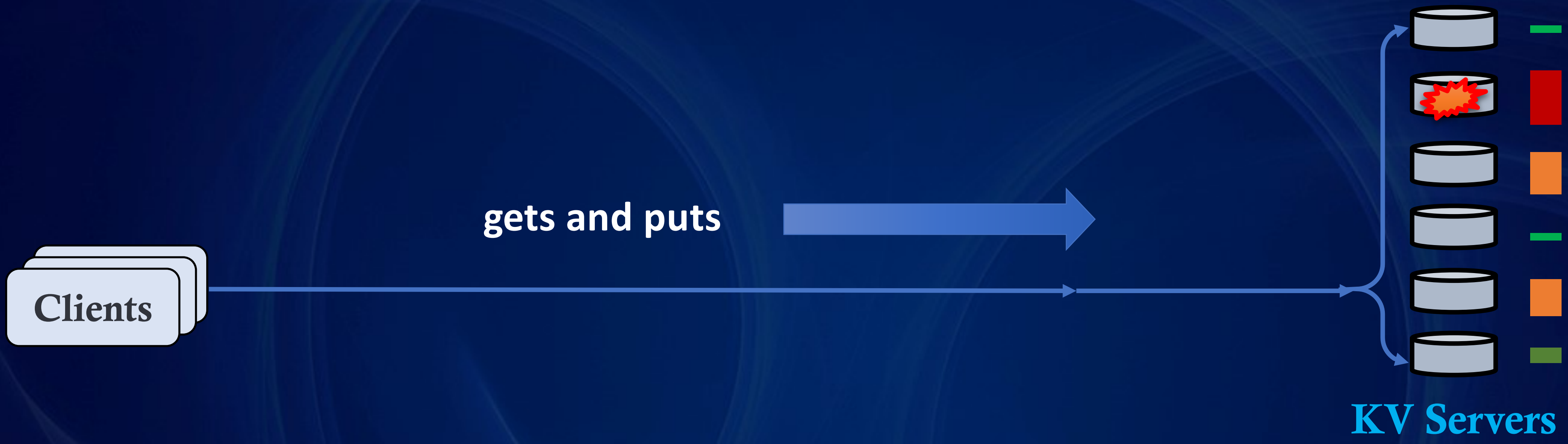
Benefits of HPCC



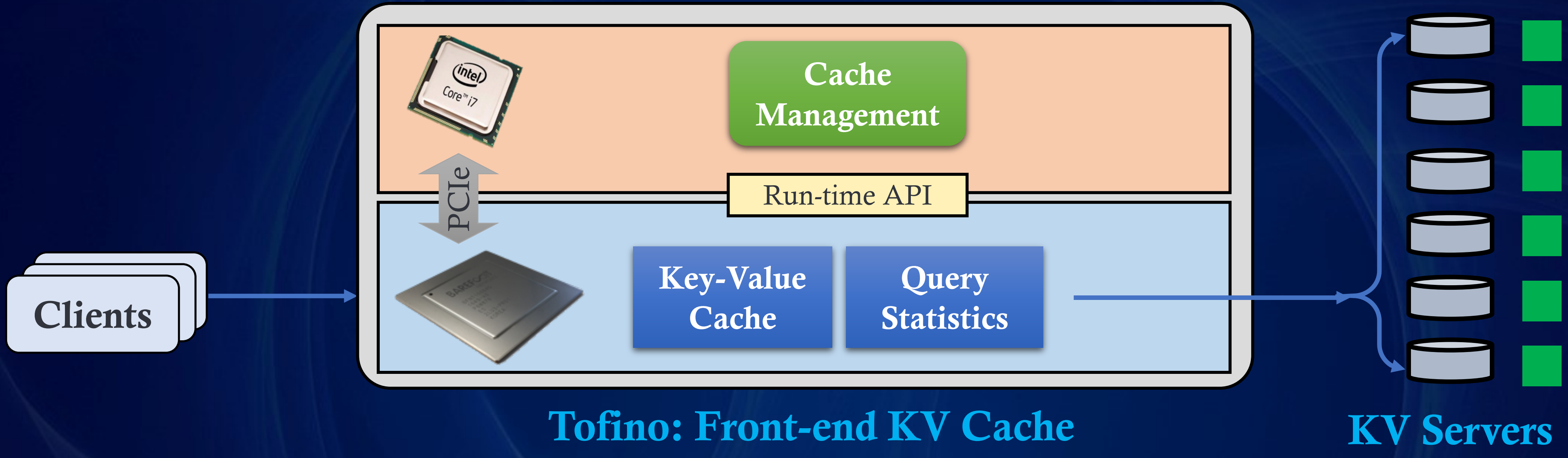


Combined Switch-NIC Architecture Leads To Innovations

NetCache: Front-end Read-only KV Cache

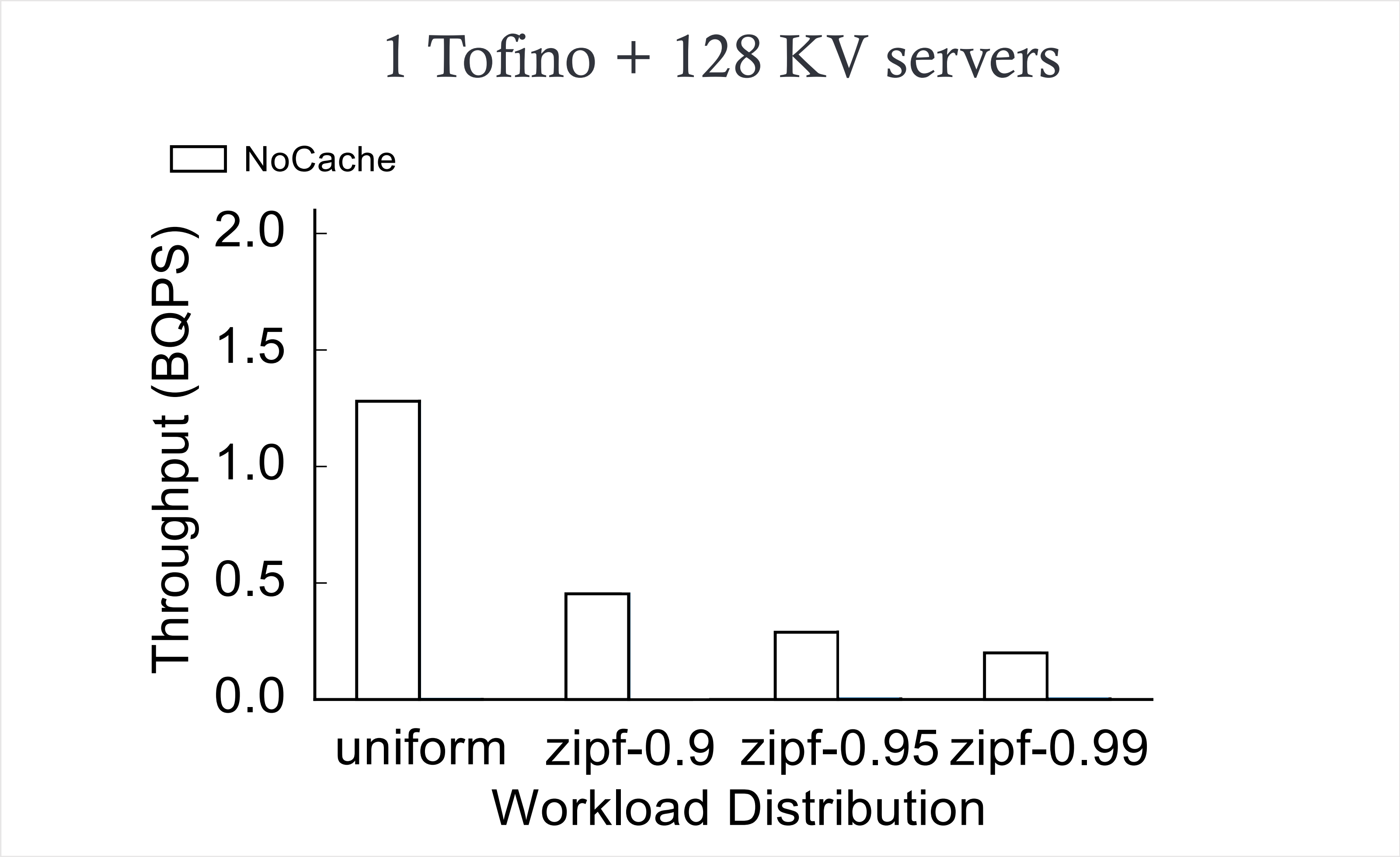


NetCache: Front-end Read-only KV Cache



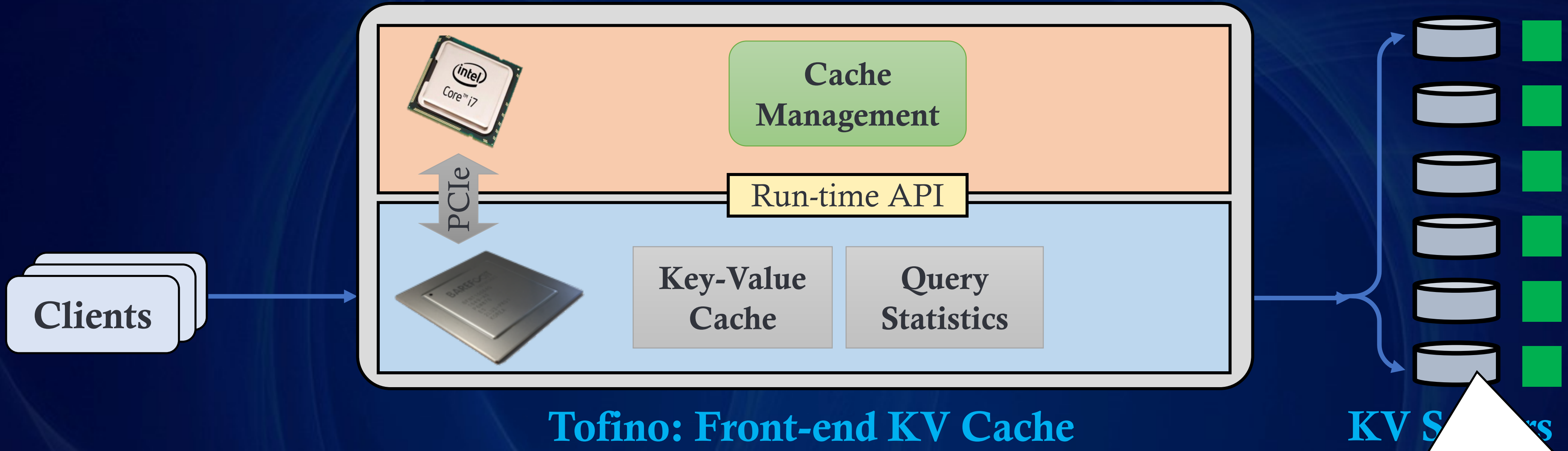
- **Data plane**
  - A very small key-value store to serve read queries for cached keys
  - Query statistics to enable efficient cache updates
- **Control plane**
  - Insert hot items into the cache and evict less popular items
  - Manage memory allocation for on-chip key-value store

Benefits of NetCache

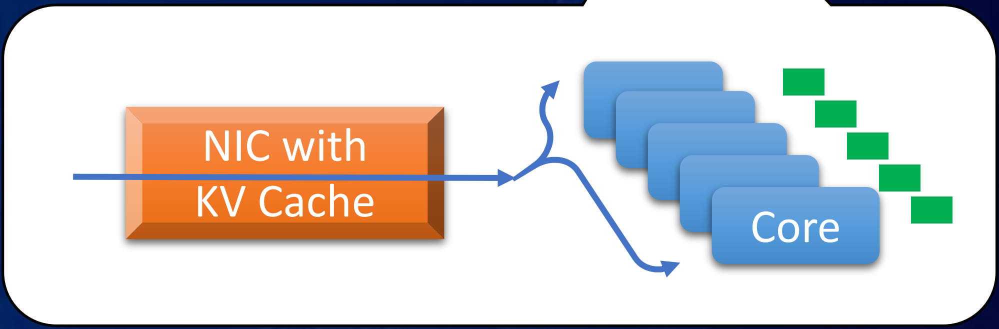


3-10x higher throughput

Natural Evolution of NetCache

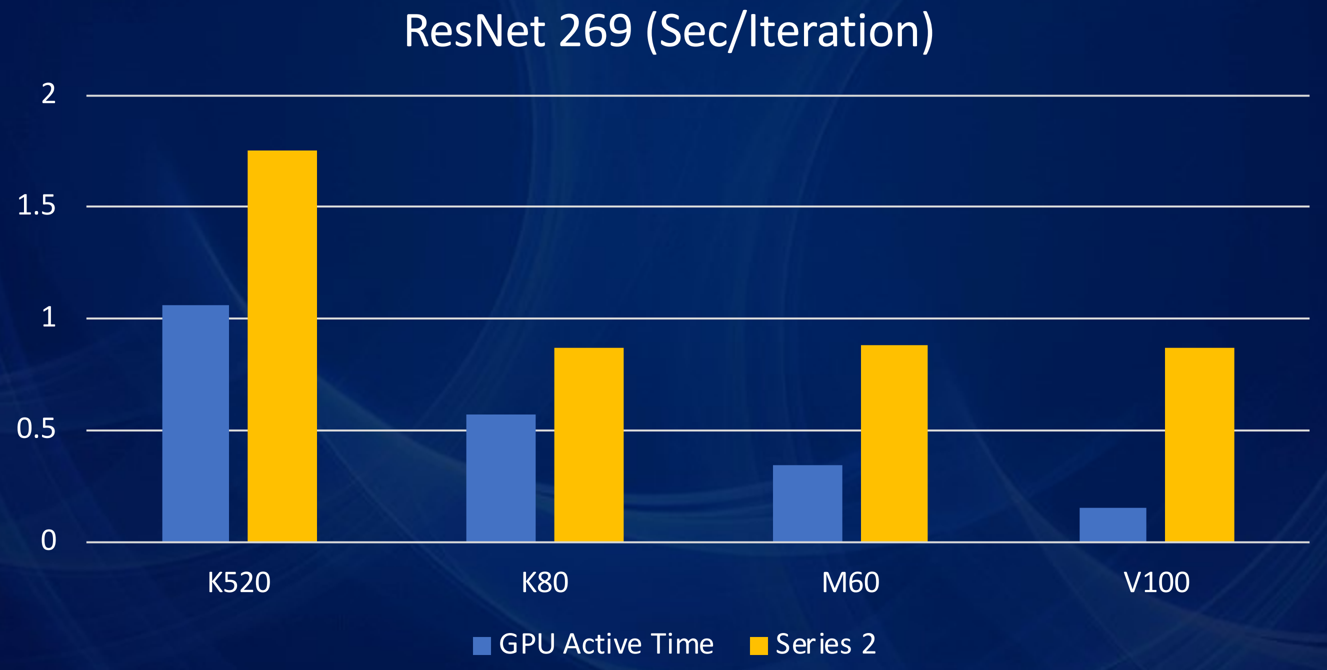


Combined Switch-NIC Architecture Leads To Innovations

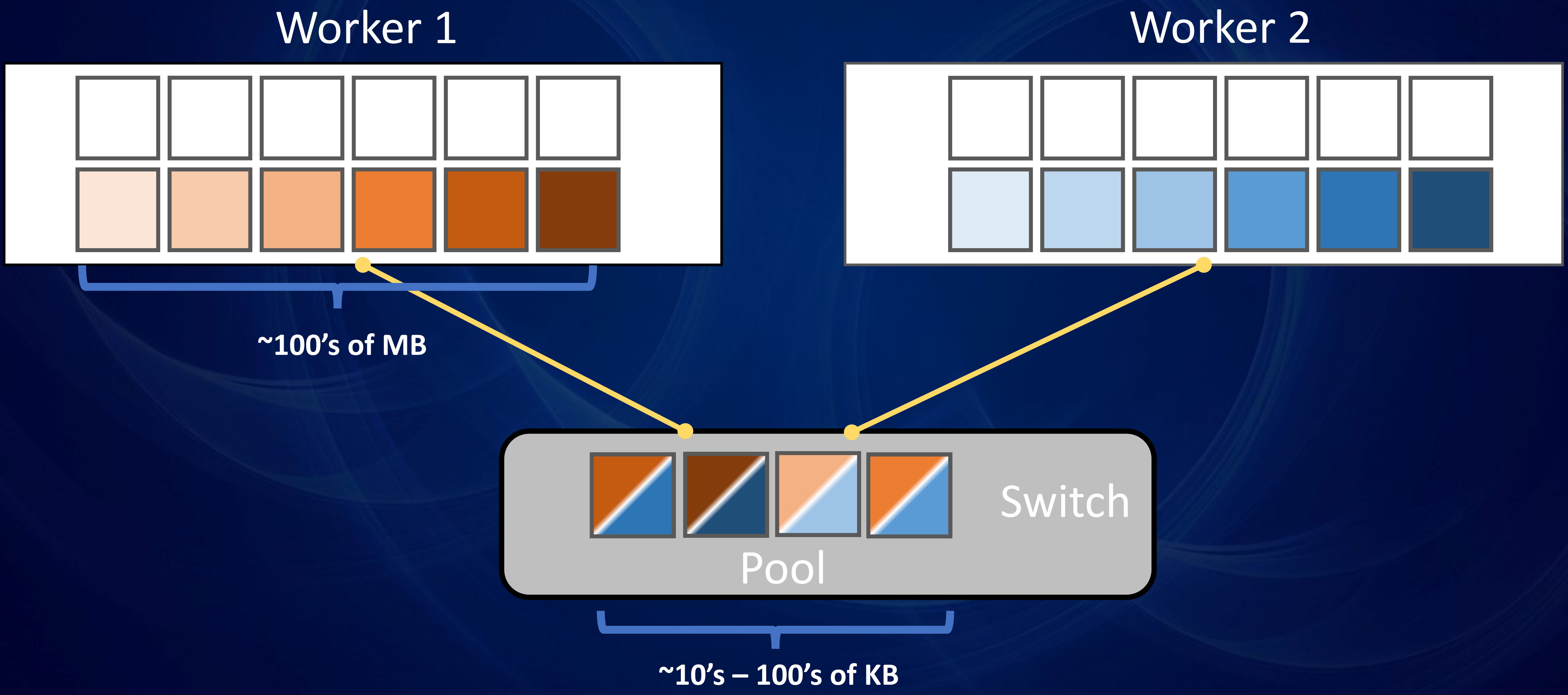


Accelerating DNN Training

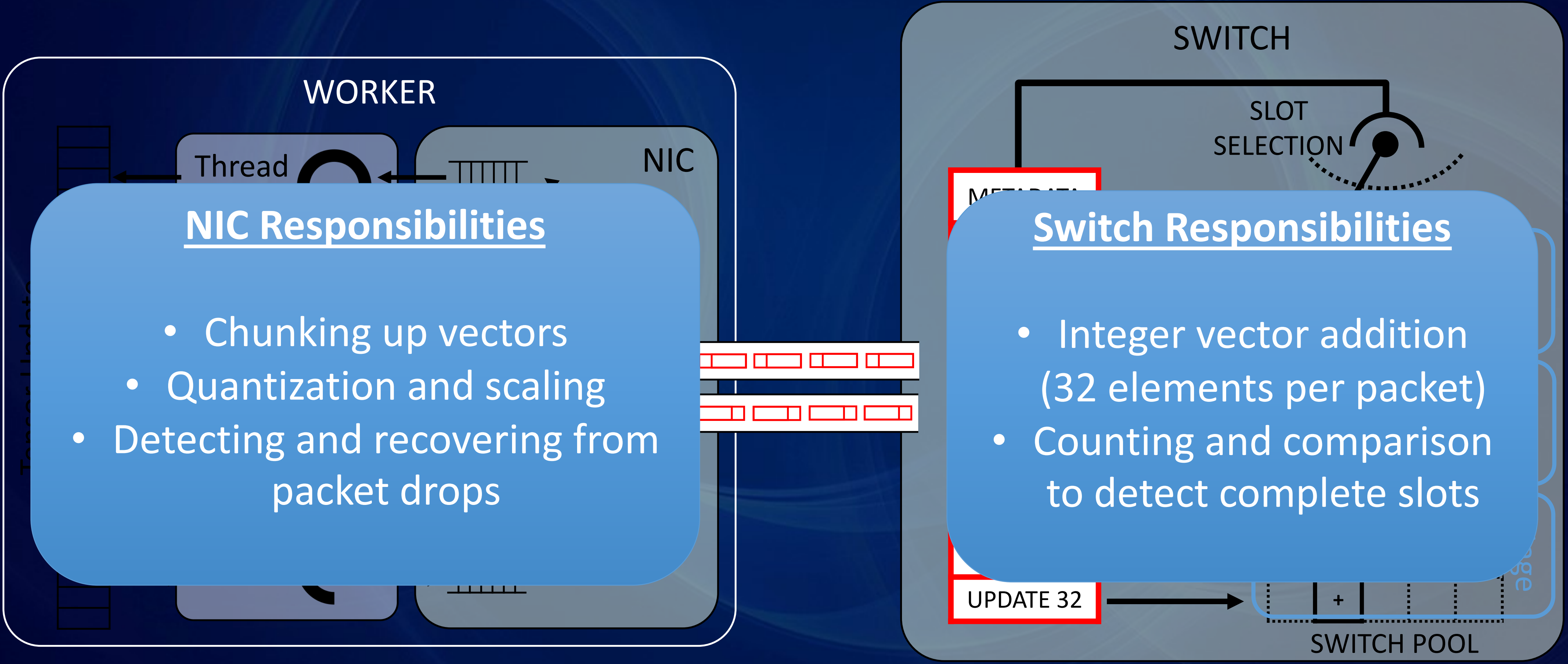
- Training over huge data requires distributed processing
- With faster workers, sharing learned parameters becomes a bottleneck



SwitchML: Streaming Aggregation via Switches



Combined Switch-NIC Architecture



NIC Responsibilities

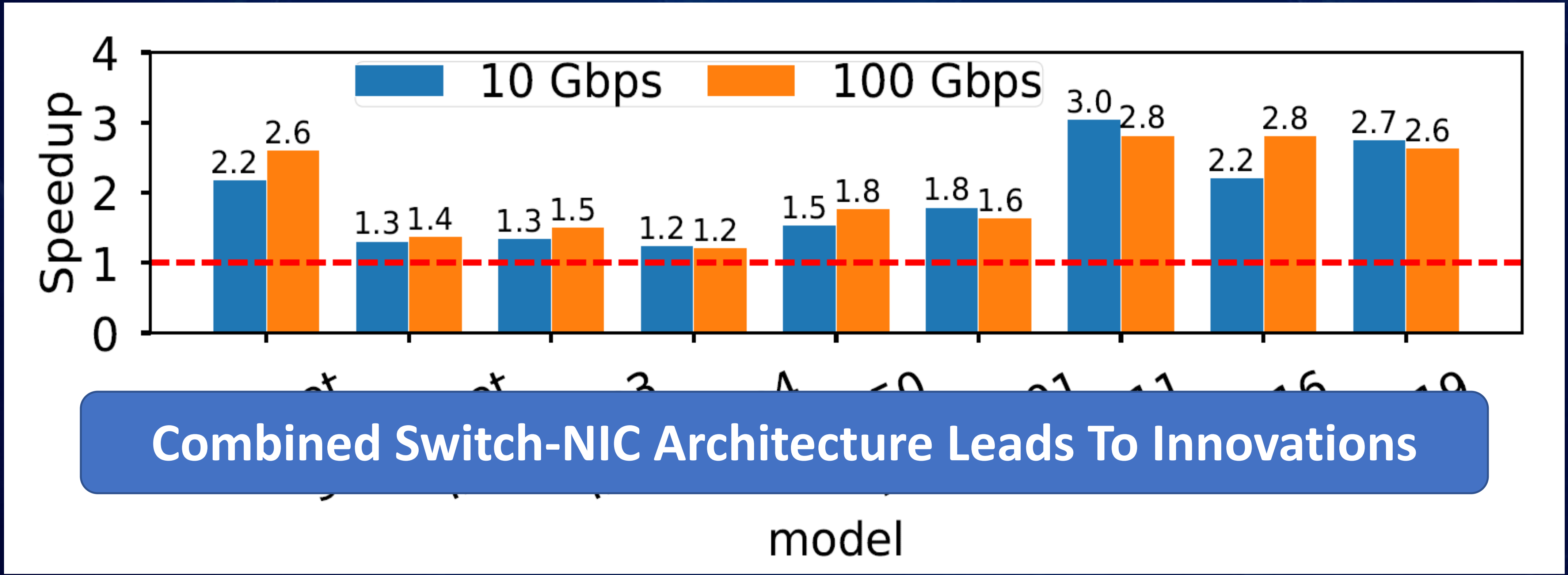
- Chunking up vectors
- Quantization and scaling
- Detecting and recovering from packet drops

Switch Responsibilities

- Integer vector addition (32 elements per packet)
- Counting and comparison to detect complete slots

How much faster is SwitchML?

SwitchML provides a speedup from **20% to 300%** compared to Tensorflow with NCCL (with direct GPU memory access)



Combined Switch-NIC Architecture Leads To Innovations

Thank you!