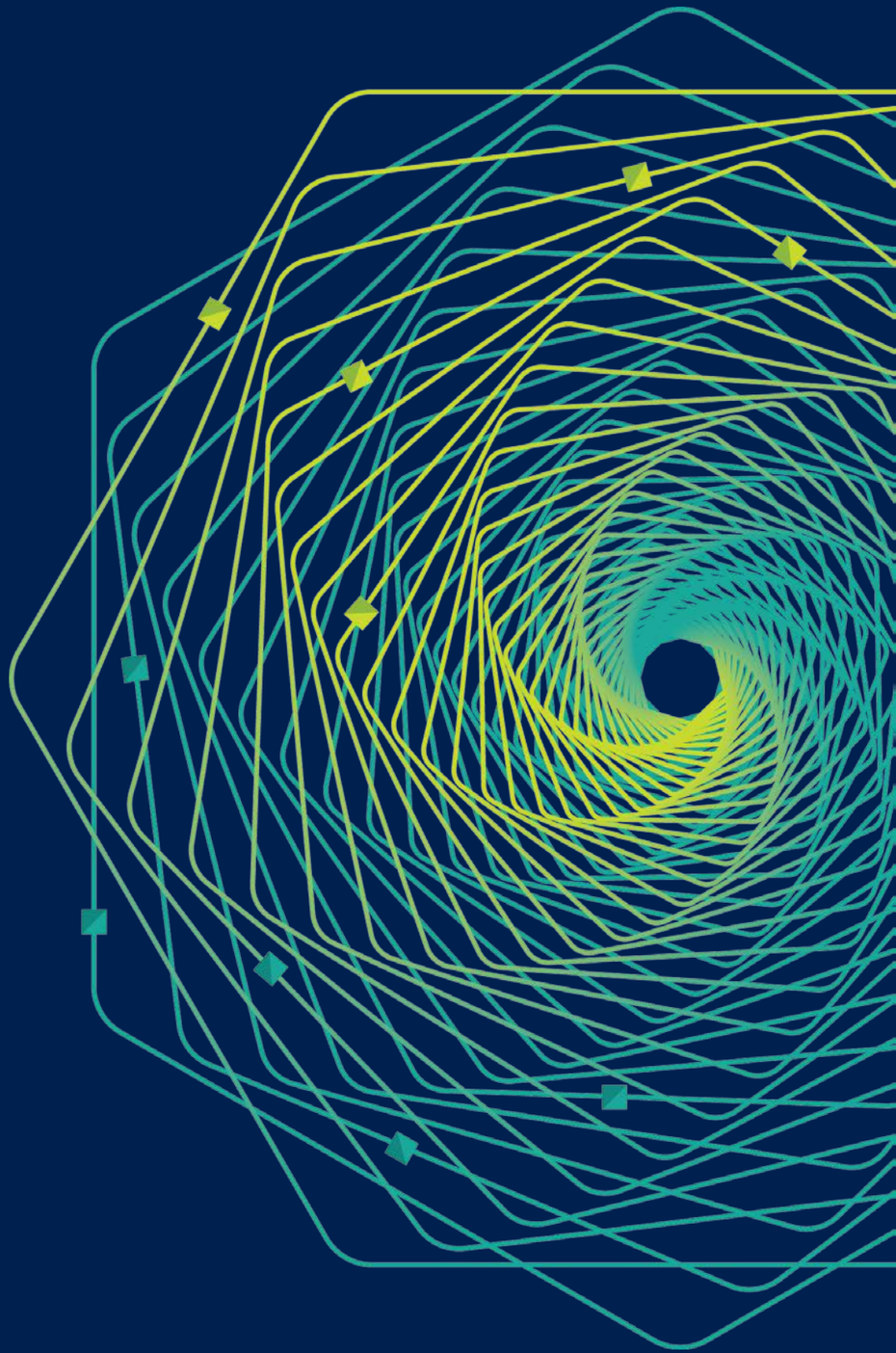
Microsoft

Research Faculty Summit 2018

Systems | Fueling future disruptions

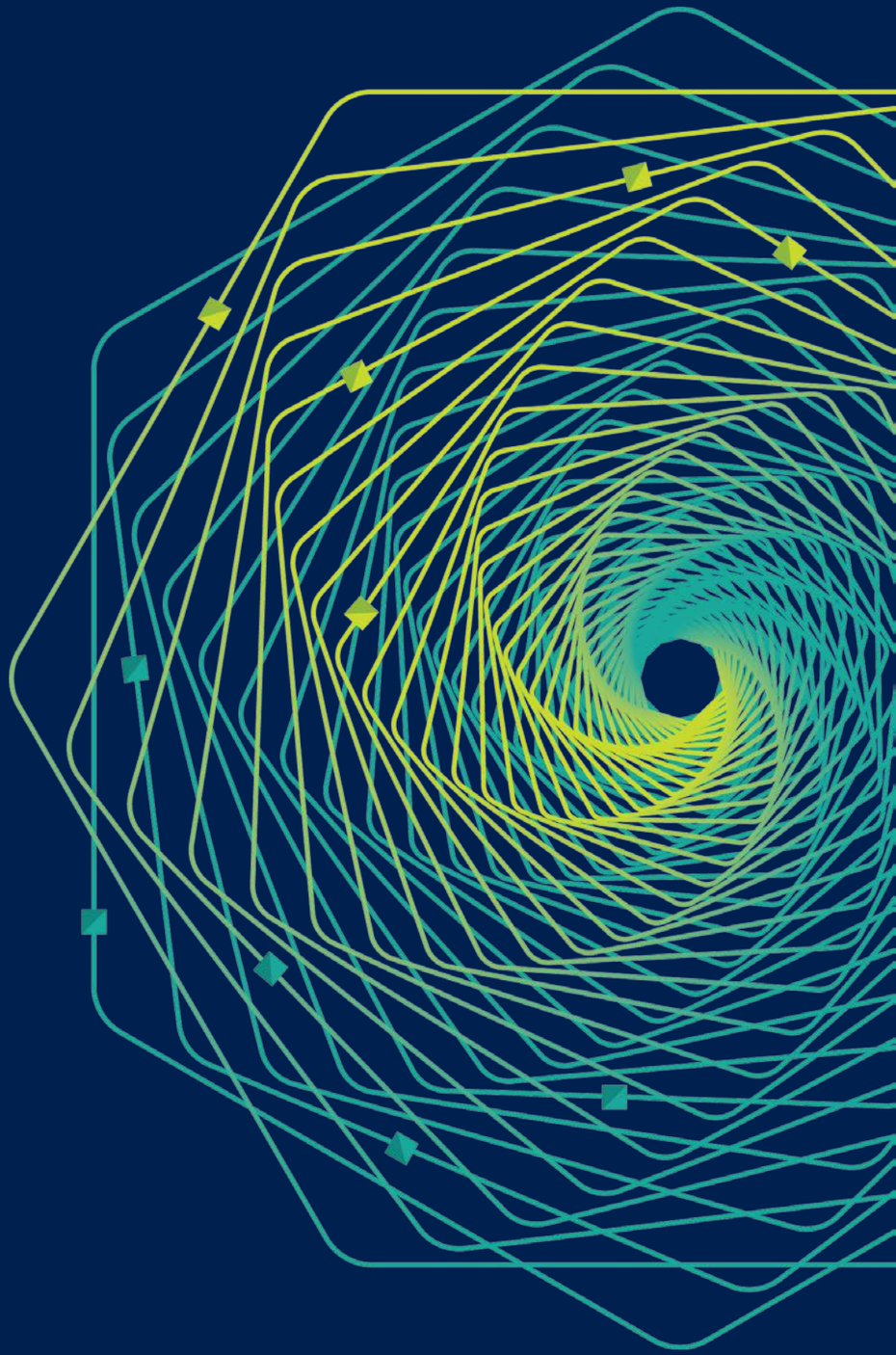


Microsoft

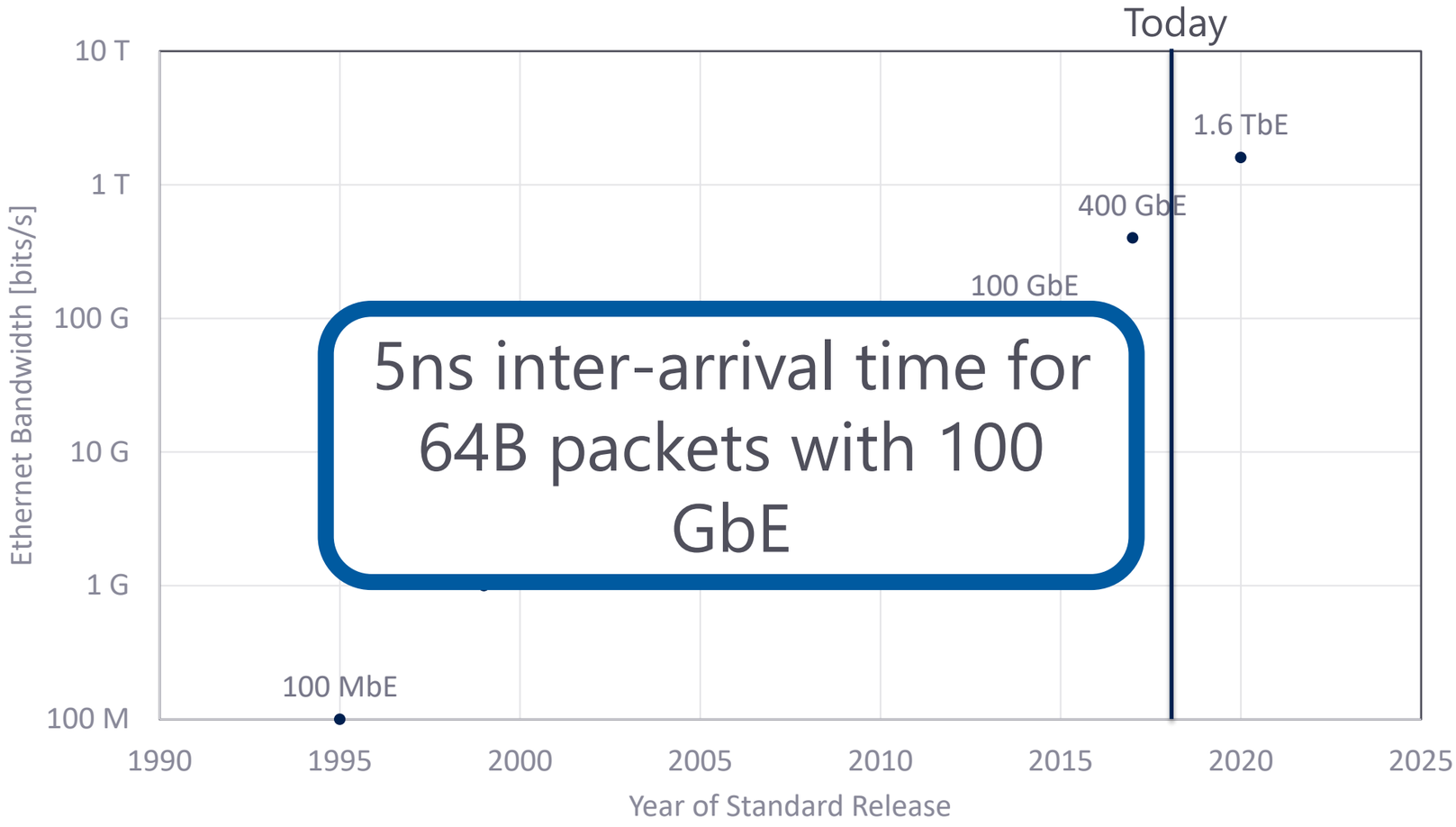
Accelerating Distributed Applications with Flexible Packet Processing

Simon Peter

Assistant Professor, UT Austin



Networks are becoming faster

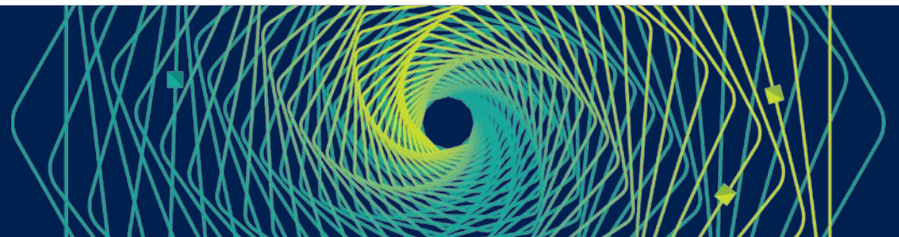


...but software packet processing is slow

- **1µs** packet echo processing time with kernel-bypass [Arrakis, OSDI'14]

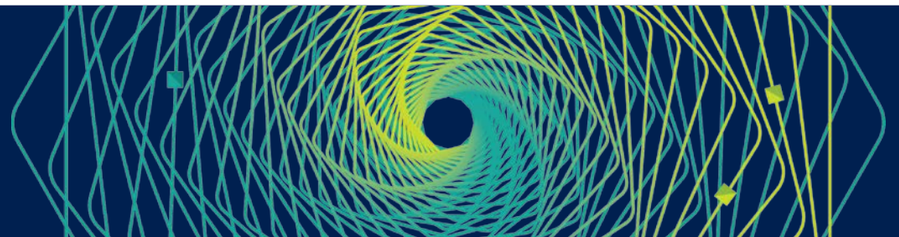
- **Single core performance has stalled**
- Parallelize? Assuming 1µs over 100Gb/s, ignoring Amdahl's Law:
  - 64B packets => 200 cores
  - 1KB packets => 14 cores

- Distributed applications are dominated by packet processing (RPCs)
  - Median message size 300B [Google, SIGCOMM'18]
  - Key-value stores, real-time analytics, file systems, ...
  - Packet processing consumes up to 90% of total processing time



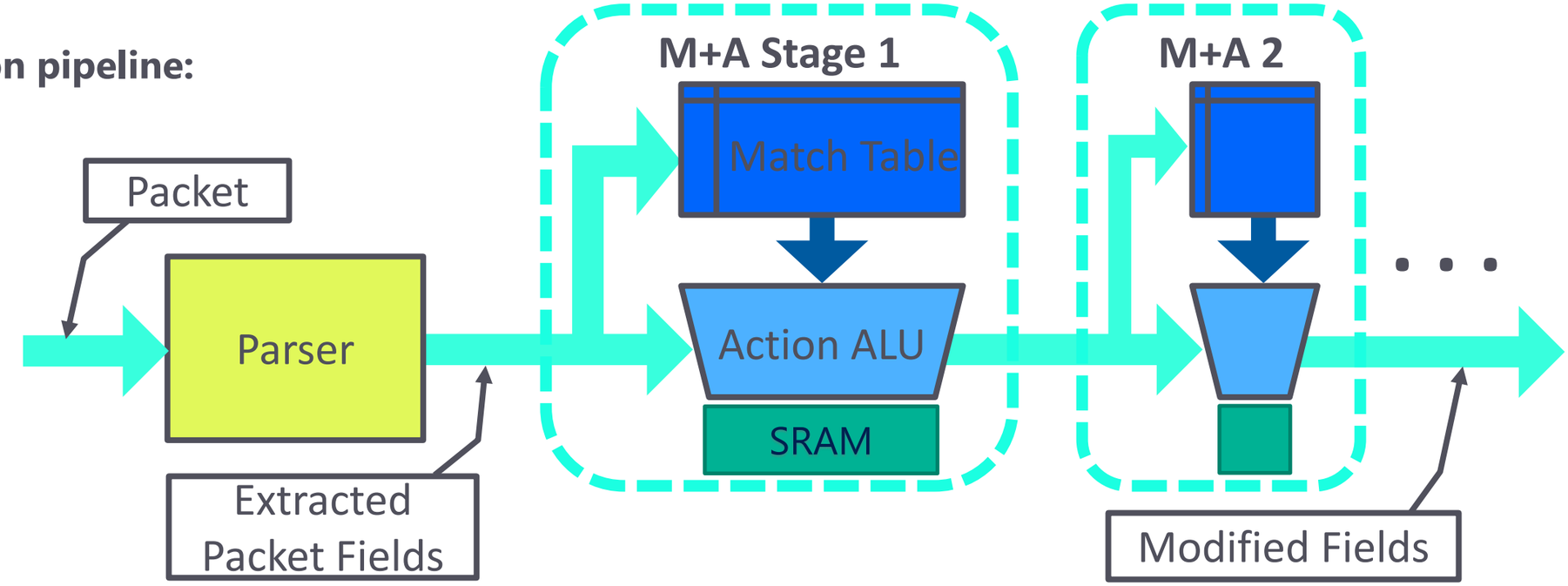
What are the alternatives?

- Fixed-function offloads: segmentation, flow director, RDMA, TCP offload engine
  - Too rigid for today's complex server & network architecture
- Full application offload to NIC (FPGA, NPU)
  - FPGA: Development is difficult—slower than software development
  - NPU: Limited per-flow performance
  - Multi-tenancy?
- **Flexible, generalized packet processing in NIC**
  - **Packet processing is systolic—use programmable hardware pipeline to offload**
  - **Define instruction set architecture for software flexibility**



FlexNIC: Flexible Packet Processing on the NIC [ASPLOS'16]

Match+action pipeline:



- Predictable latency & throughput, performance isolated multi-tenant sharing at high utilization
- Scales to 100K+ active flows/server
- FPGA implementation 32 Gb/s (Azure [NSDI'18]), ASIC implementation 640 Gb/s (RMT [SIGCOMM'13])
- Multi-pipeline switch 3.2 Tb/s (Barefoot Tofino)

Match+Action Programs (cf. P4 [SIGCOMM'14])

Supports:

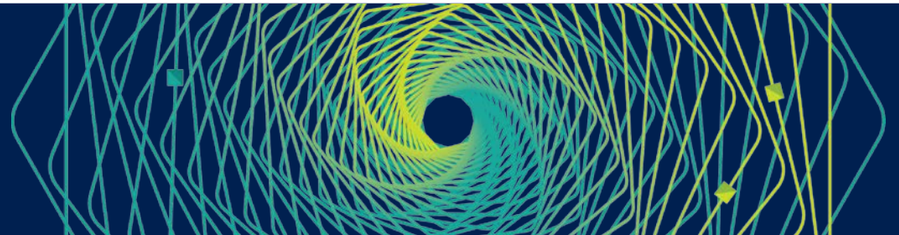
- Steer/filter/modify packets
- Calculate hash/checksum
- Trigger reply packets

Does not support:

- Complex calculations
- Loops
- Arbitrary state

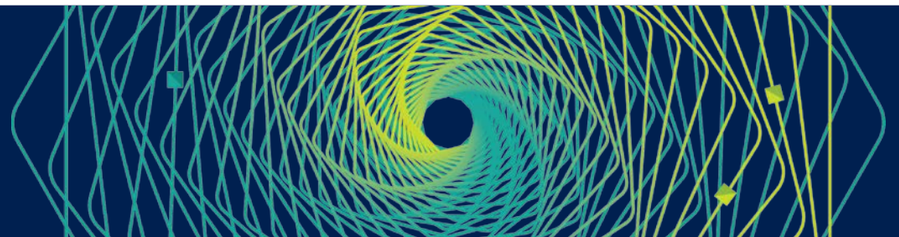
Example: Tuple-to-worker steering in real-time analytics

```
Match:
    IF tcp.dstport == STORM_PORT && is_tuple(tcp.storm)
Action:
    DMA tcp.storm.tuple TO cores[tcp.storm.worker]
    ACK(tcp.srcport)
```

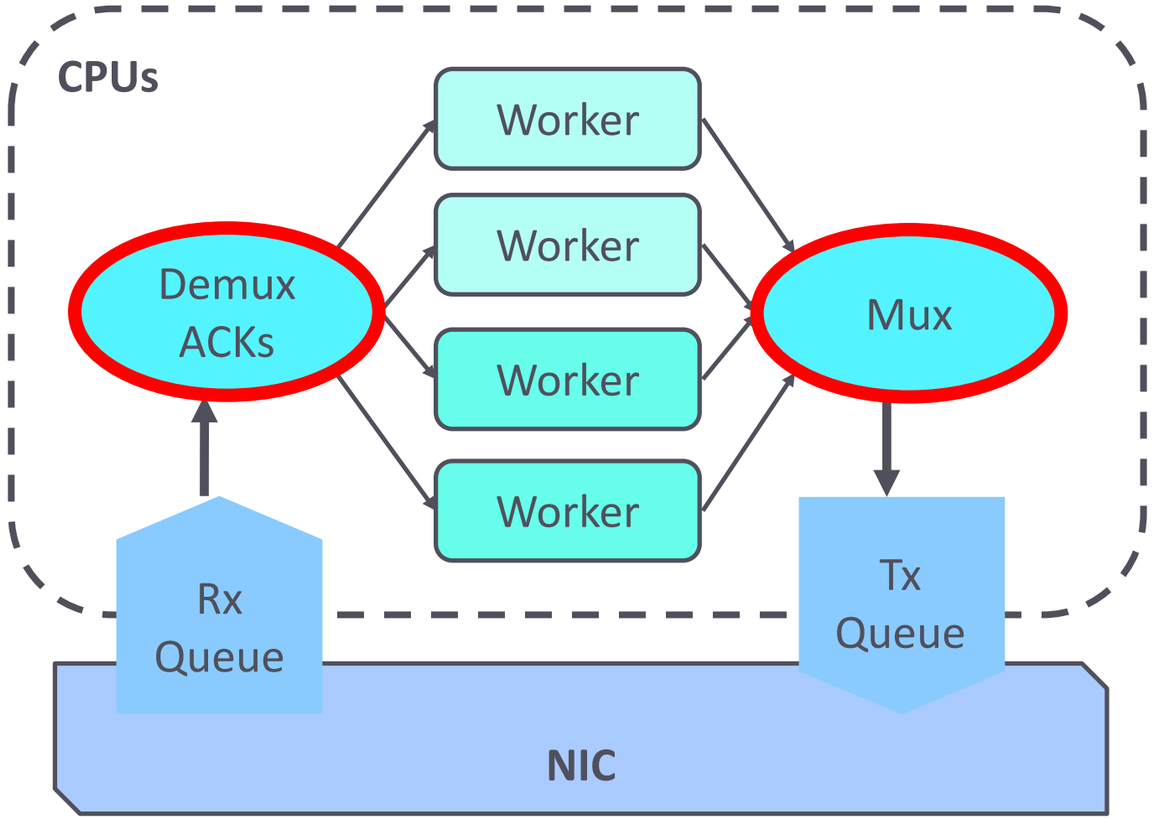
Use Cases

- **Real-time analytics**
  - **Offload tuple-to-worker steering, acknowledgment generation**
- Key-value store, network intrusion detection, other NFs
  - Smart steering for better cache utilization
  - E.g., key-based steering for KV store
- Distributed file systems (with non-volatile memory)
  - Replication protocol on NIC improves tail-latency (Hyperloop [SIGCOMM'18])
  - Common case of NFS/SMB server data plane on NIC
- Web servers
  - QUIC on NIC with client-adaptive congestion control
- Distributed consensus (Paxos), vSwitch, caching, TCP, …



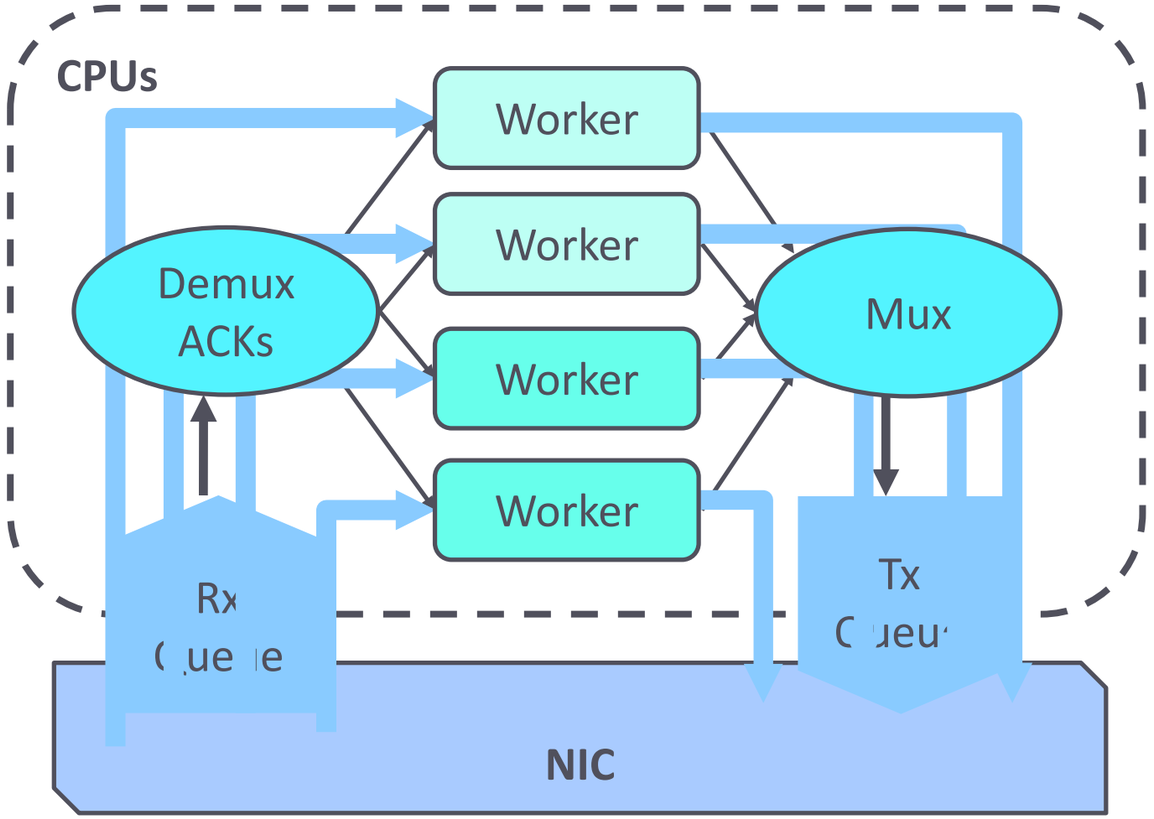
Use Case: Real-time Analytics

- Data stream processing
  - Replicated workers run on CPUs
  - Dataset updates (tuples) stream through
- Workers communicate all-all
  - Doing so per core is too expensive
  - **Instead:**
    Multiplex onto per-machine connections
- (De-)Multiplexing threads are performance bottleneck
  - 2 CPUs required for 10 Gb/s
    => 20 CPUs for 100 Gb/s



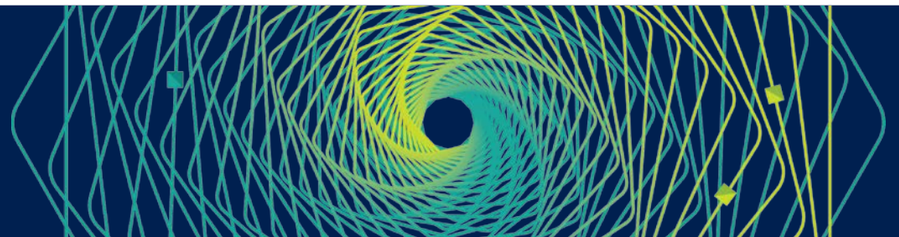
Use Case: Real-time Analytics

- Offload (de)multiplexing and ACK generation to FlexNIC
  - Demux based on steering table
  - ACKs sent back on same connection
- CPUs free to run workers
  - 3x throughput improvement on 10GbE



Summary

- Networks are becoming faster, CPUs are not
  - Distributed applications require all available CPUs
- **FlexNIC:** A NIC architecture for flexible packet processing offload
  - Packet processing is systolic—can be offloaded to programmable match+action pipeline
  - Predictable latency & throughput, sharing semantics, scalability
- Many use cases
  - Real-time analytics—offload multiplexing, acknowledgment generation
  - Key-value storage, TCP, distributed file systems, consensus, vSwitch, caching, …



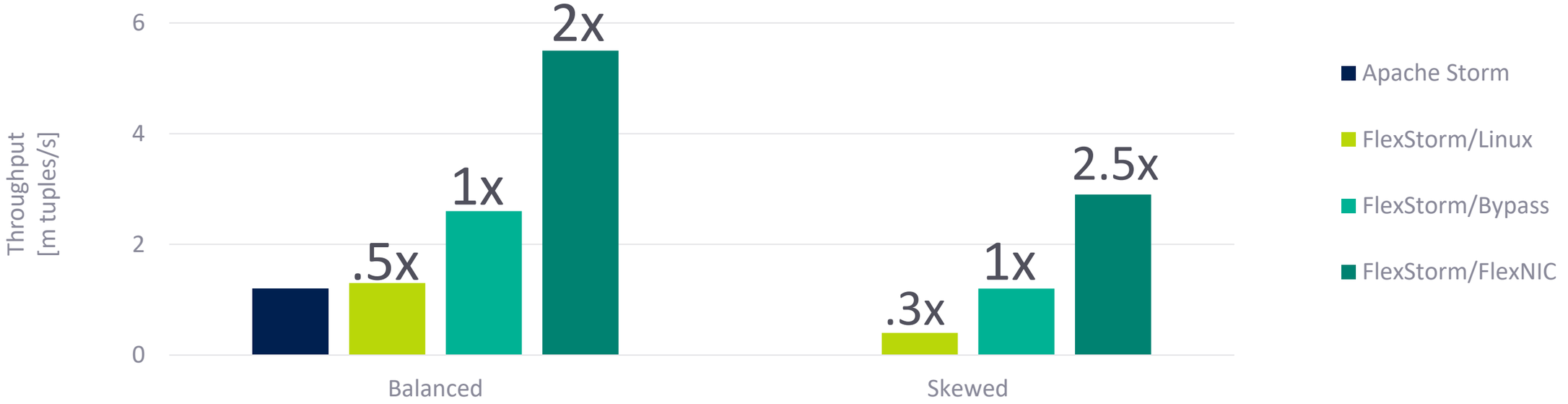
Thank you!

Microsoft

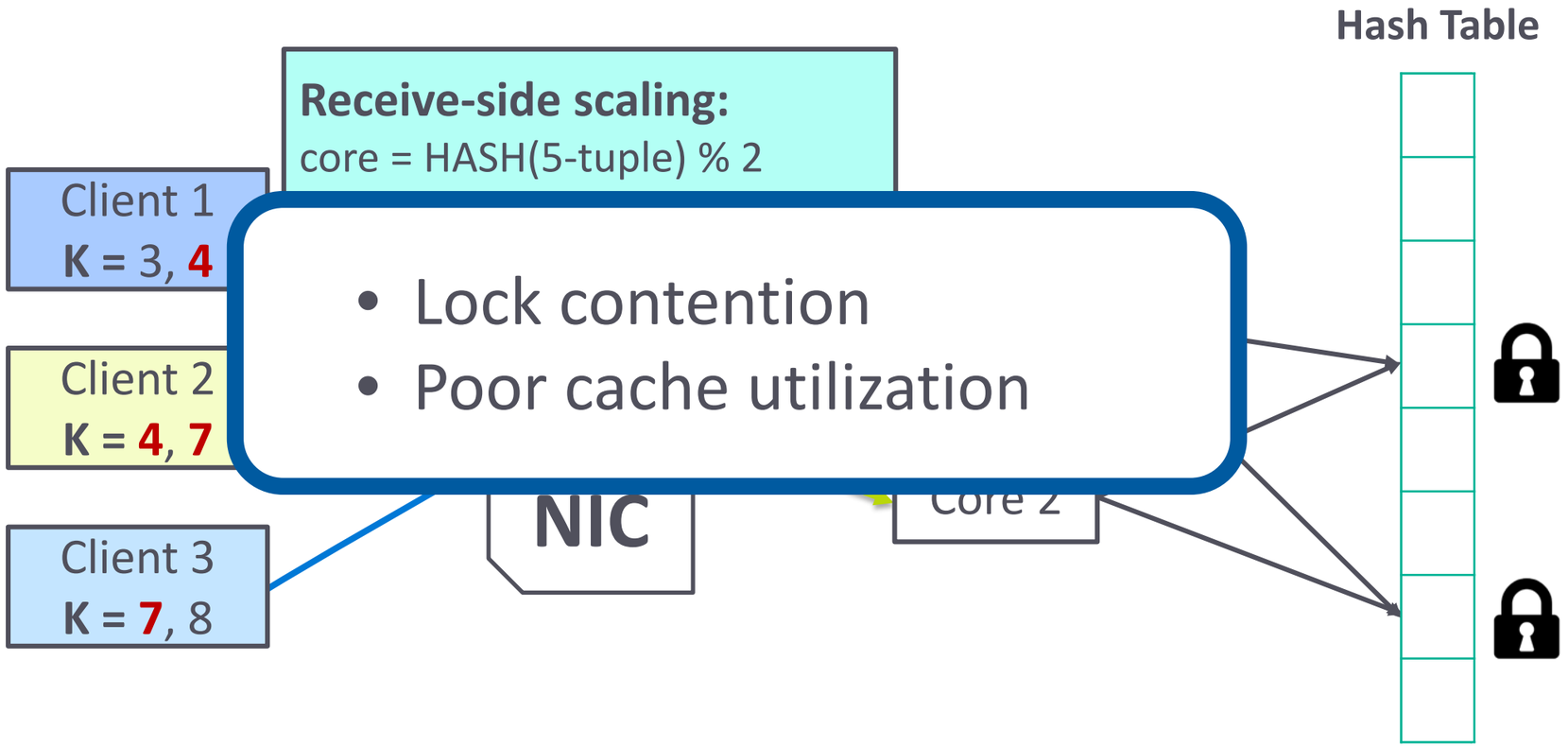
© Copyright Microsoft Corporation. All rights reserved.

Real-time Analytics Performance Evaluation

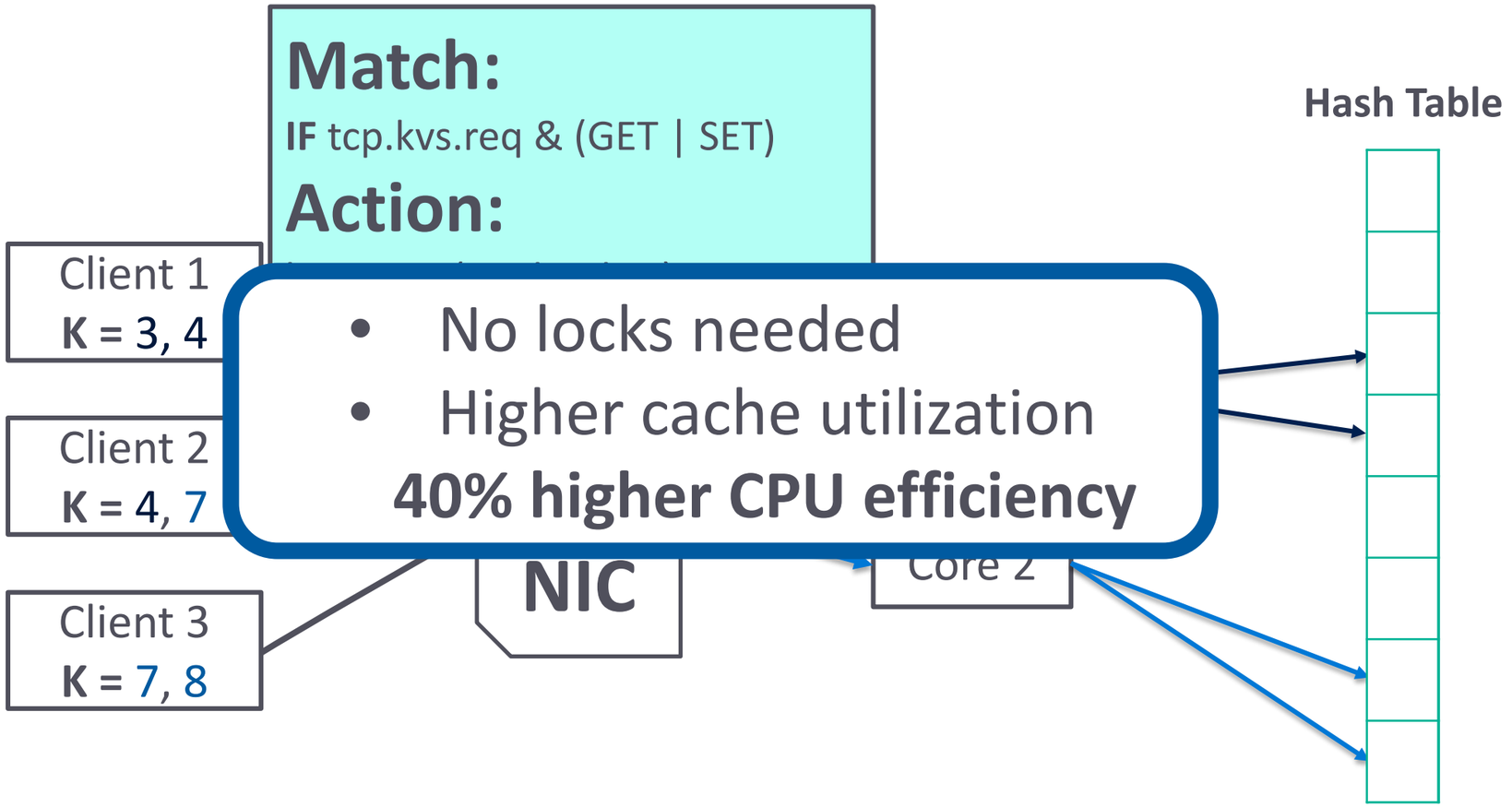
- Cluster of 3 machines (6 cores, 10Gb/s NICs)
  - Balanced: Balanced network load
  - Skewed: One server is bottleneck



Key-value Store with fixed-function RSS



Key-value Store with key-based steering



Match+Action Programs (cf. P4 [SIGCOMM'14])

Supports:

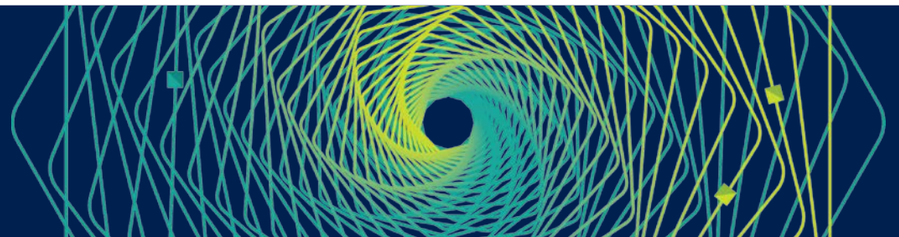
- Steer/filter/modify packets
- Calculate hash/checksum
- Trigger reply packets

Does not support:

- Complex calculations
  - Loops
  - Arbitrary state

Example: Key-based request steering in a key-value store

```
Match:
   IF tcp.dstport == KVS_PORT && (tcp.kvs.req & (GET | SET))
Action:
   hash = HASH(tcp.kvs.key)
   DMA hash, tcp.kvs TO cores[hash % NCORES]
   TCP_ACK(tcp.srcport)
```

Optimize packet processing – How and where?

- Optimize packet processing on CPUs? [SoftNIC (Berkeley'15), TCP onload (IEEE'09)]
  - Not enough – takes CPUs away from applications
  - CPU energy-efficiency << FPGA << ASIC
- Packet processing is systolic
  - Can be accelerated with domain-specific hardware pipeline
- NIC is perfectly situated – sees all traffic
  - Has to forward packets among host CPUs and network anyway
  - Great place to enforce OS network policy