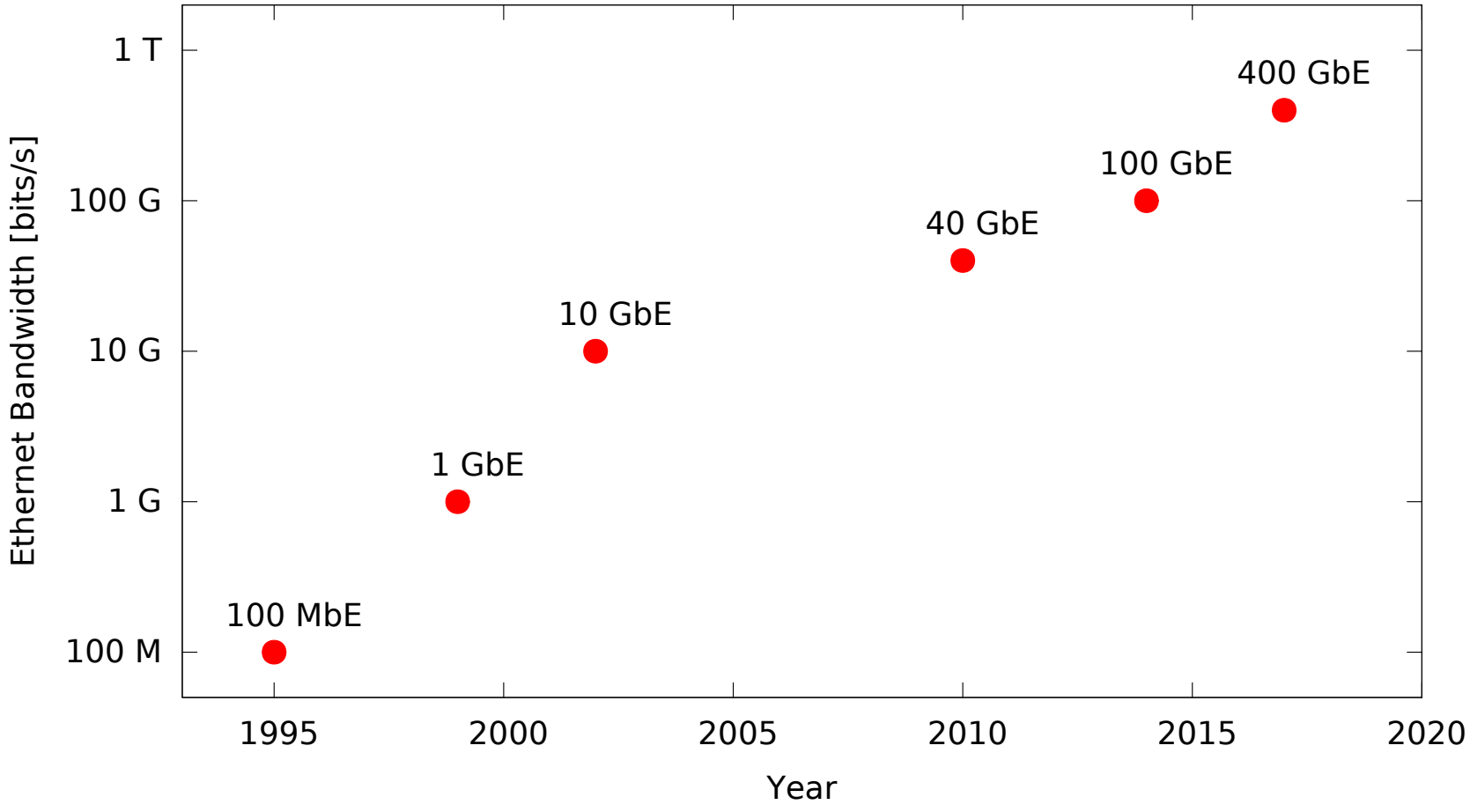
FlexNIC: Rethinking Network DMA

Antoine Kaufmann Simon Peter
Tom Anderson Arvind Krishnamurthy

University of Washington

HotOS 2015

Networks: Fast and Growing Faster



- 5ns inter-arrival time for 64B packets at 100Gbps

Software needs to catch up

- Many cloud apps dominated by packet processing
  - Key-value stores, graph analytics, load balancers

Software needs to catch up

- Many cloud apps dominated by packet processing
  - Key-value stores, graph analytics, load balancers
- Redis on Arrakis with kernel-bypass: $4\mu s$
  - Still a long way to go to 5ns

Software needs to catch up

- Many cloud apps dominated by packet processing
  - Key-value stores, graph analytics, load balancers
- Redis on Arrakis with kernel-bypass: $4\mu s$
  - Still a long way to go to 5ns
- 5ns is not a lot of time
  - Cache access: 15ns for L3, 40ns if dirty in other L1

Software needs to catch up

- Many cloud apps dominated by packet processing
  - Key-value stores, graph analytics, load balancers
- Redis on Arrakis with kernel-bypass: $4\mu s$
  - Still a long way to go to 5ns
- 5ns is not a lot of time
  - Cache access: 15ns for L3, 40ns if dirty in other L1
- **Careful memory system use is paramount**
  - Ideal: Data always in L1/L2 cache

NIC & SW are not well integrated

- Poor cache locality, extra synchronization
  - NIC steers packets to cores by connection
  - Application locality may not match connection
- Wasted CPU cycles
  - Packet parsing repeated in software
- High memory system pressure
  - Packet formatted for network, not SW access
  - High packet processing overhead

Our Proposal: FlexNIC

- Flexible NIC DMA interface
  - Applications insert packet matching rules
  - Rules control DMA actions

Our Proposal: FlexNIC

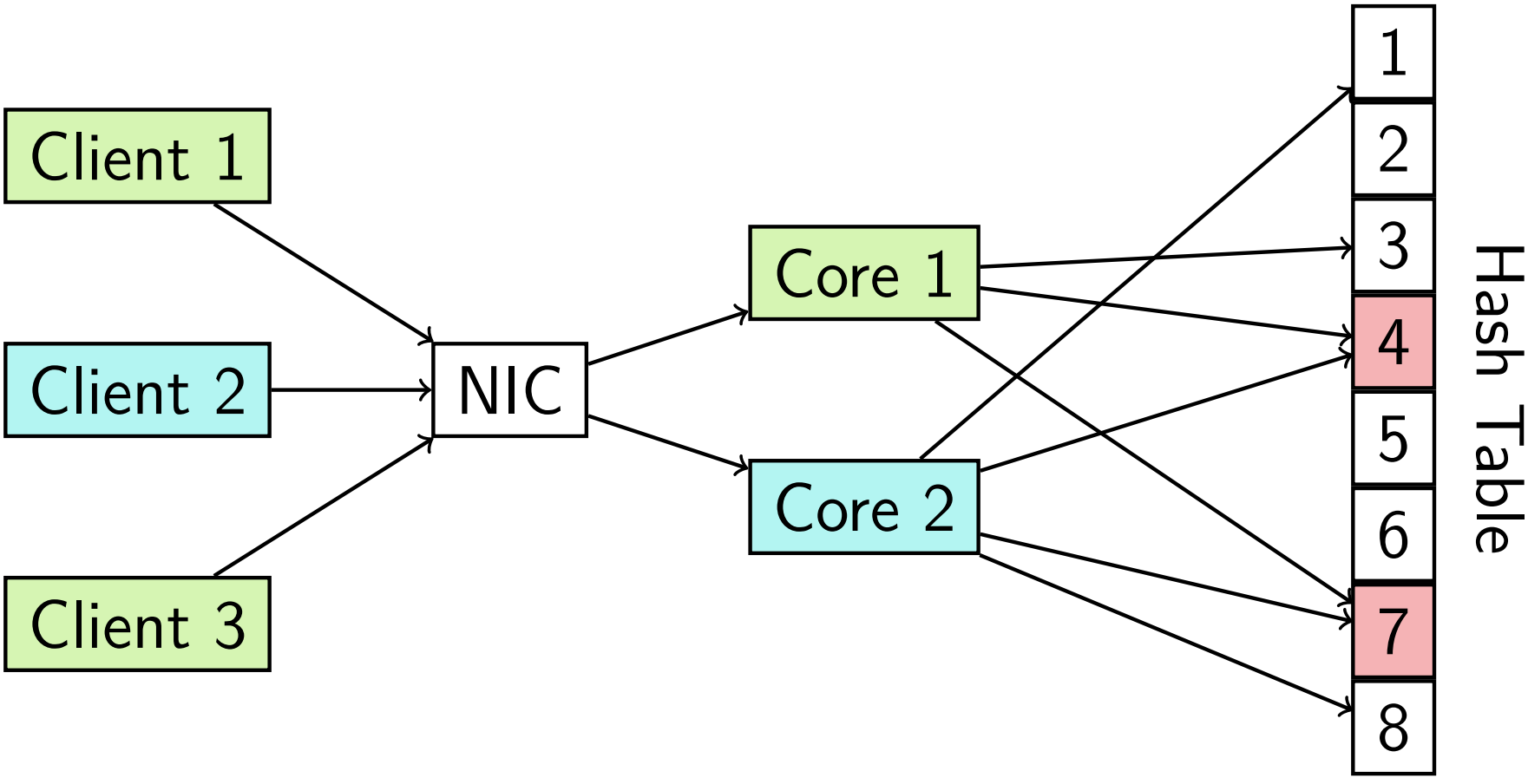
- Flexible NIC DMA interface
  - Applications insert packet matching rules
  - Rules control DMA actions
- Use multi-stage match+action (M+A) processing
  - Similar to that found in next-generation SDN switches

Our Proposal: FlexNIC

- Flexible NIC DMA interface
  - Applications insert packet matching rules
  - Rules control DMA actions
- Use multi-stage match+action (M+A) processing
  - Similar to that found in next-generation SDN switches
- M+A is both efficient and a flexible abstraction
  - Packet **steering** based on app-defined match
  - App-level packet **validation**
  - Customized packet **transformations**: add/remove/modify header fields
  - Can be stateful

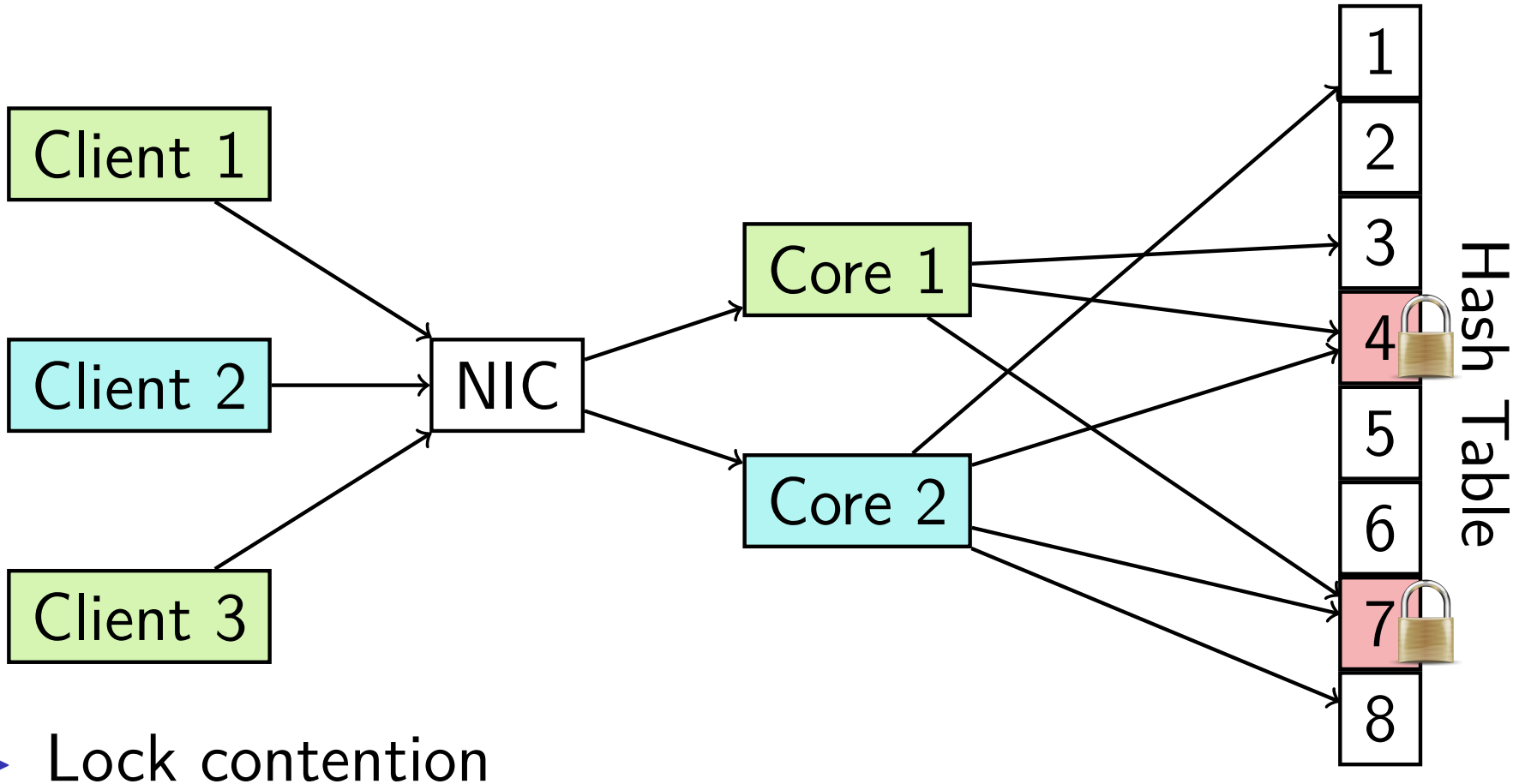
Example: Key-Value Store

Receive-Side Scaling



Example: Key-Value Store

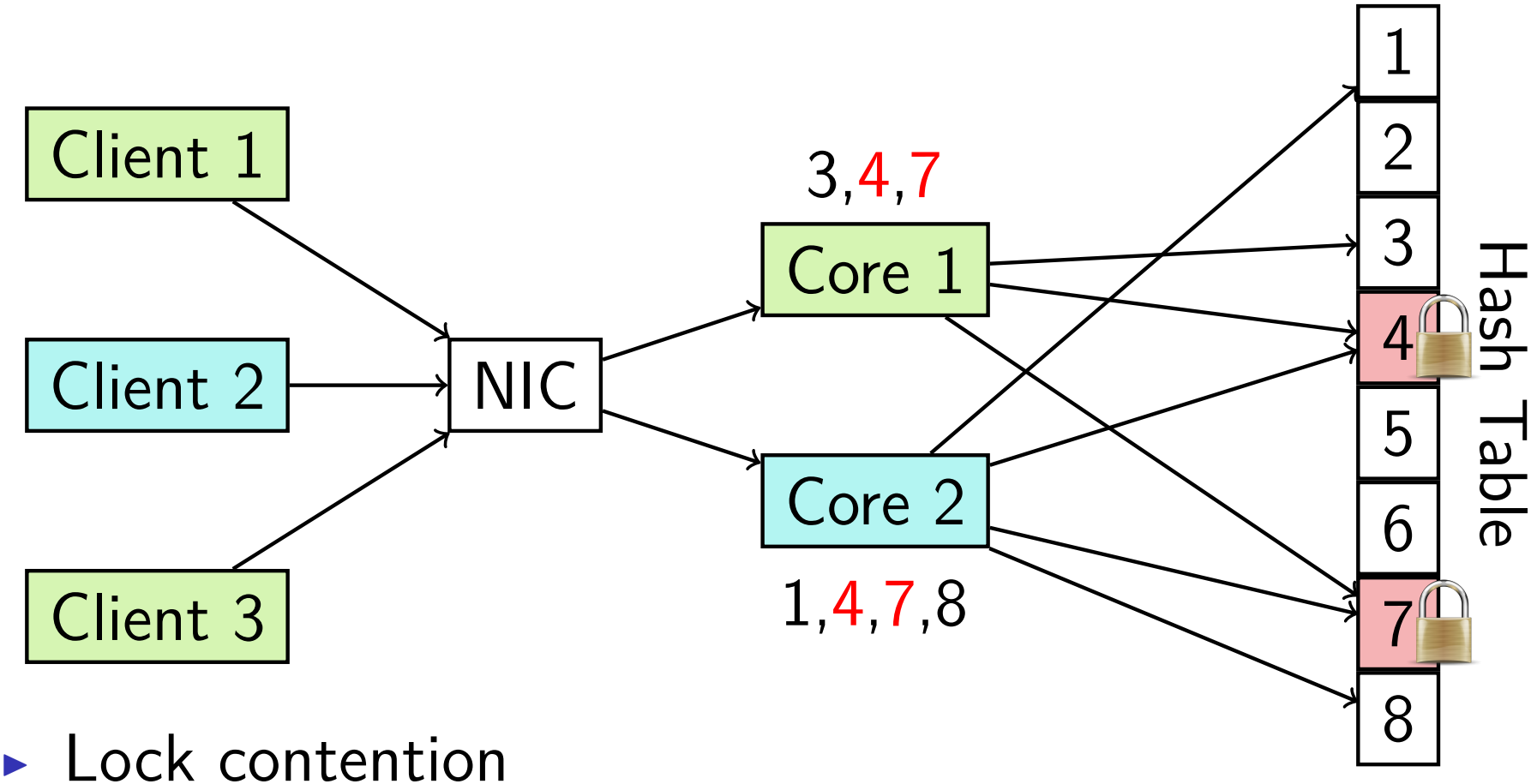
Receive-Side Scaling



- Lock contention

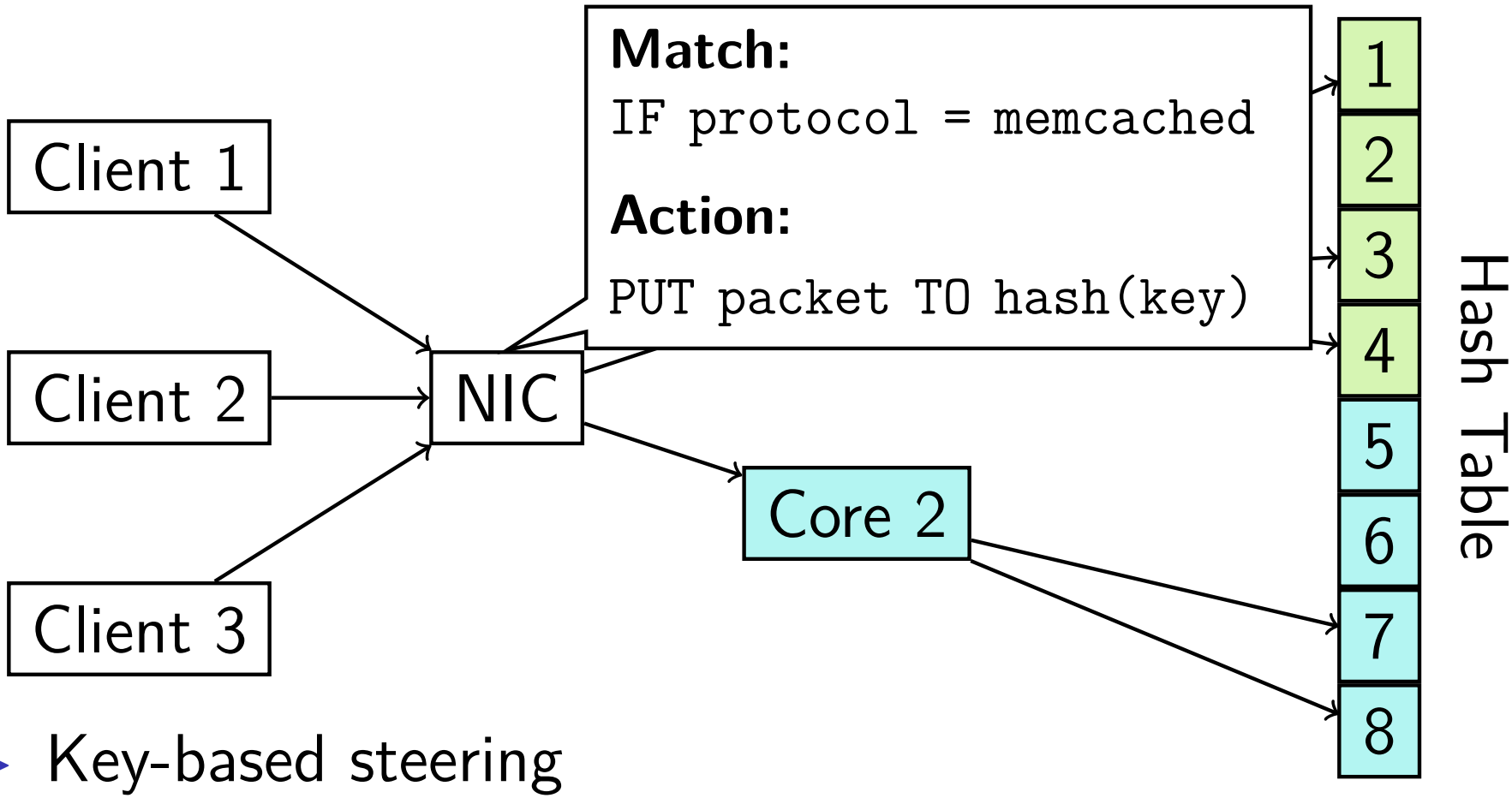
Example: Key-Value Store

Receive-Side Scaling



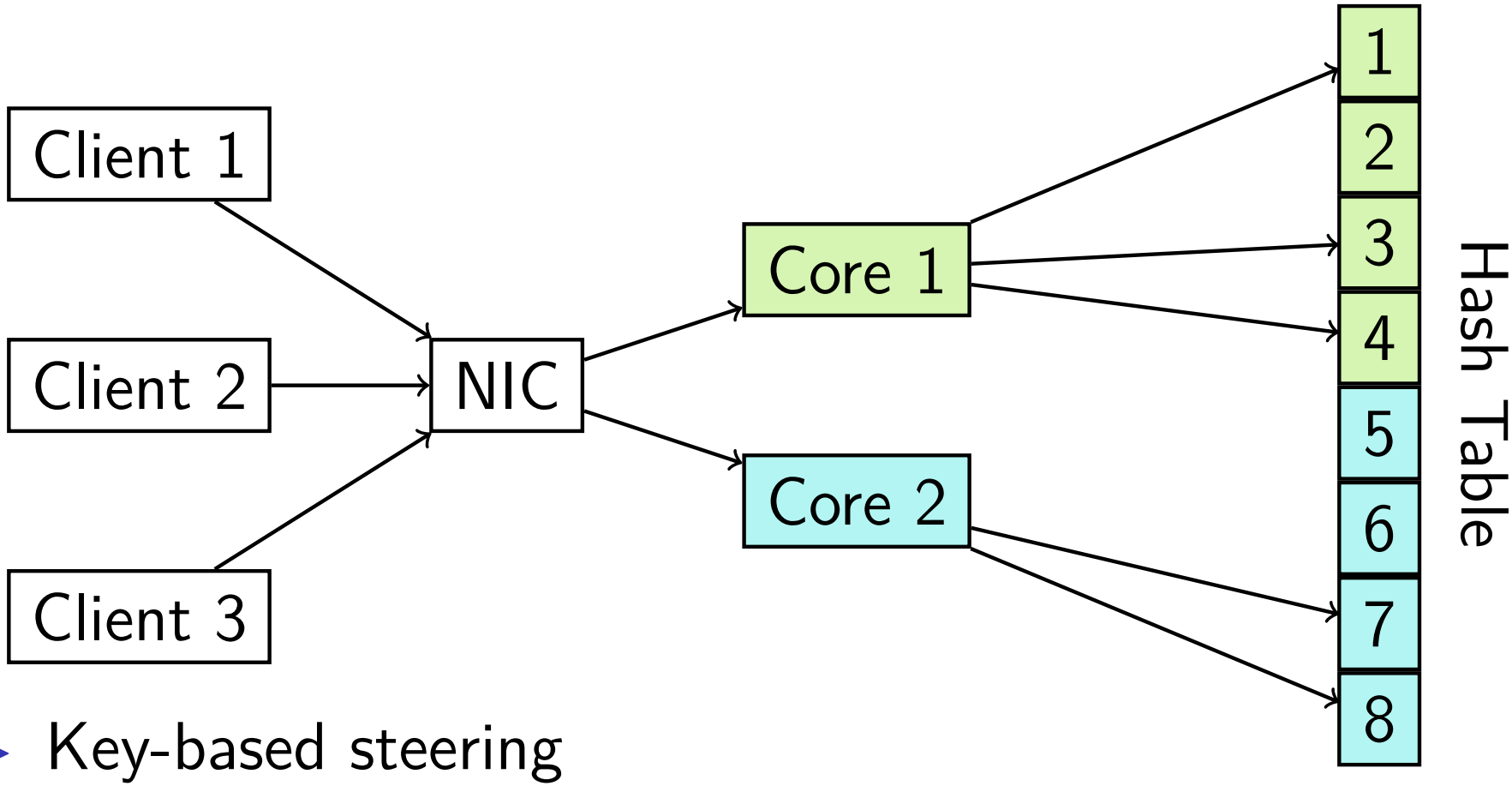
- Lock contention
- Poor cache utilization

FlexNIC Steering



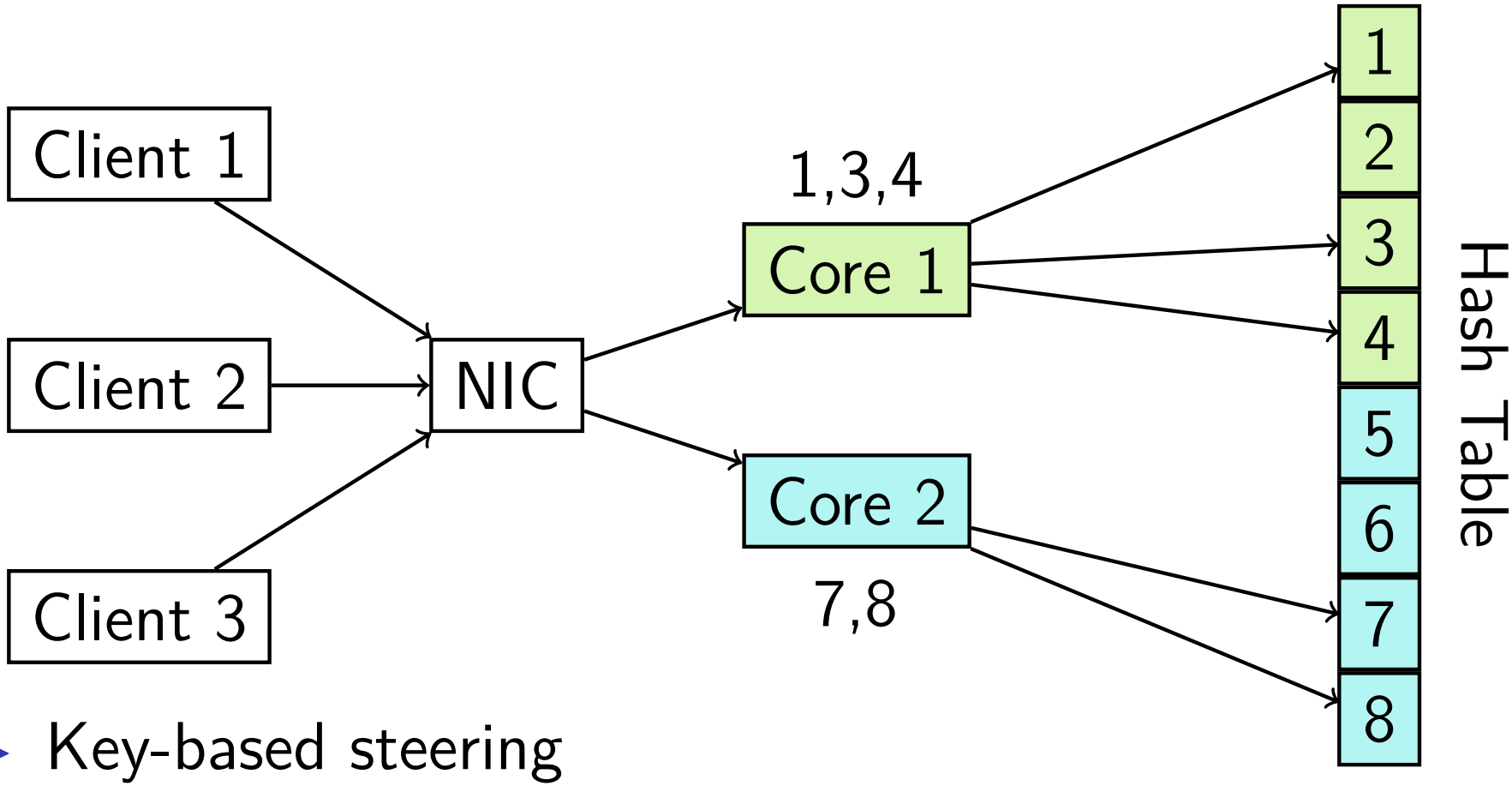
- Key-based steering

FlexNIC Steering



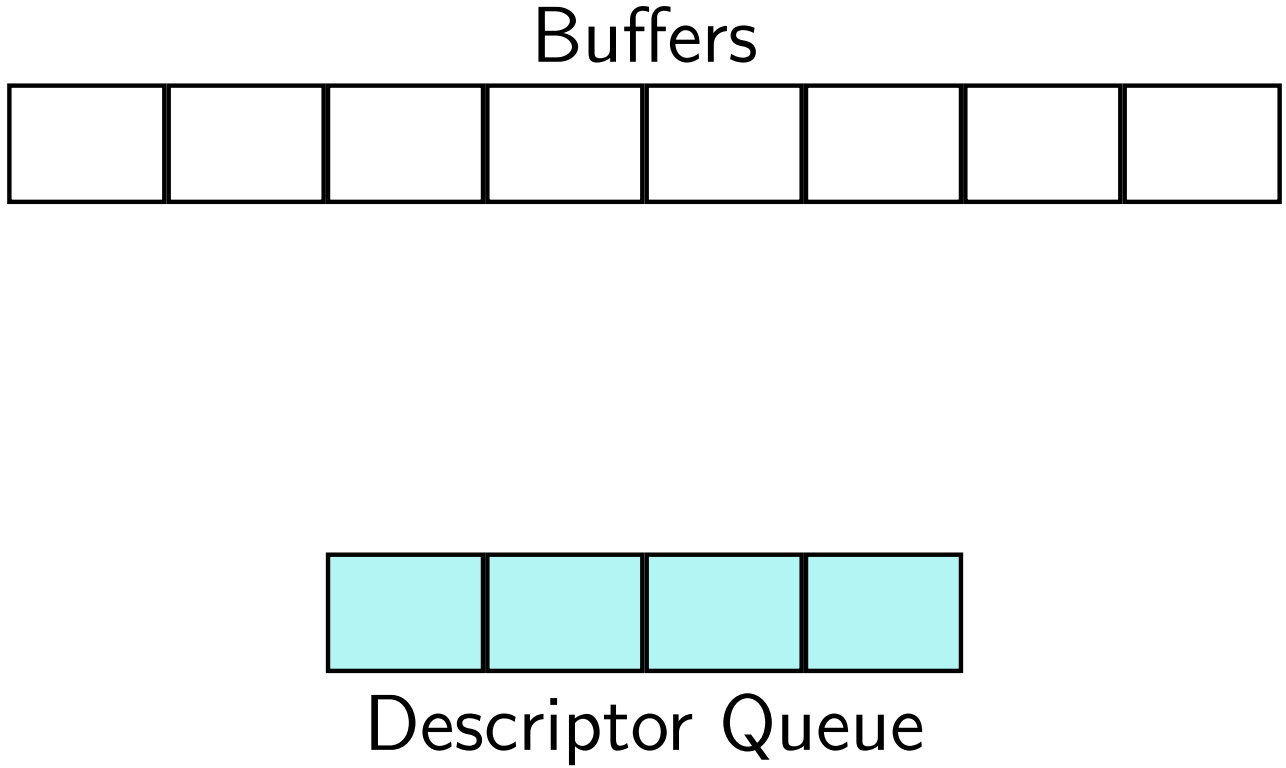
- Key-based steering
  - No locks needed

FlexNIC Steering

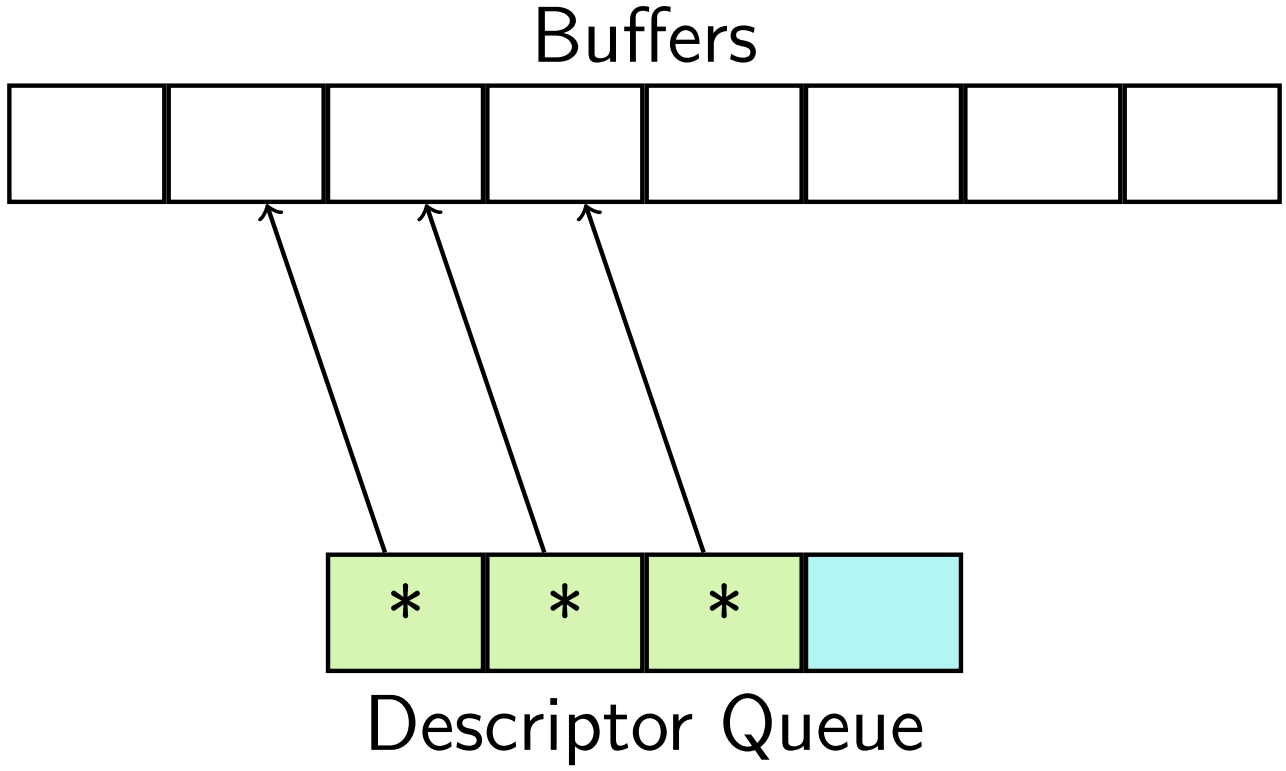


- Key-based steering
  - No locks needed
  - Better cache utilization

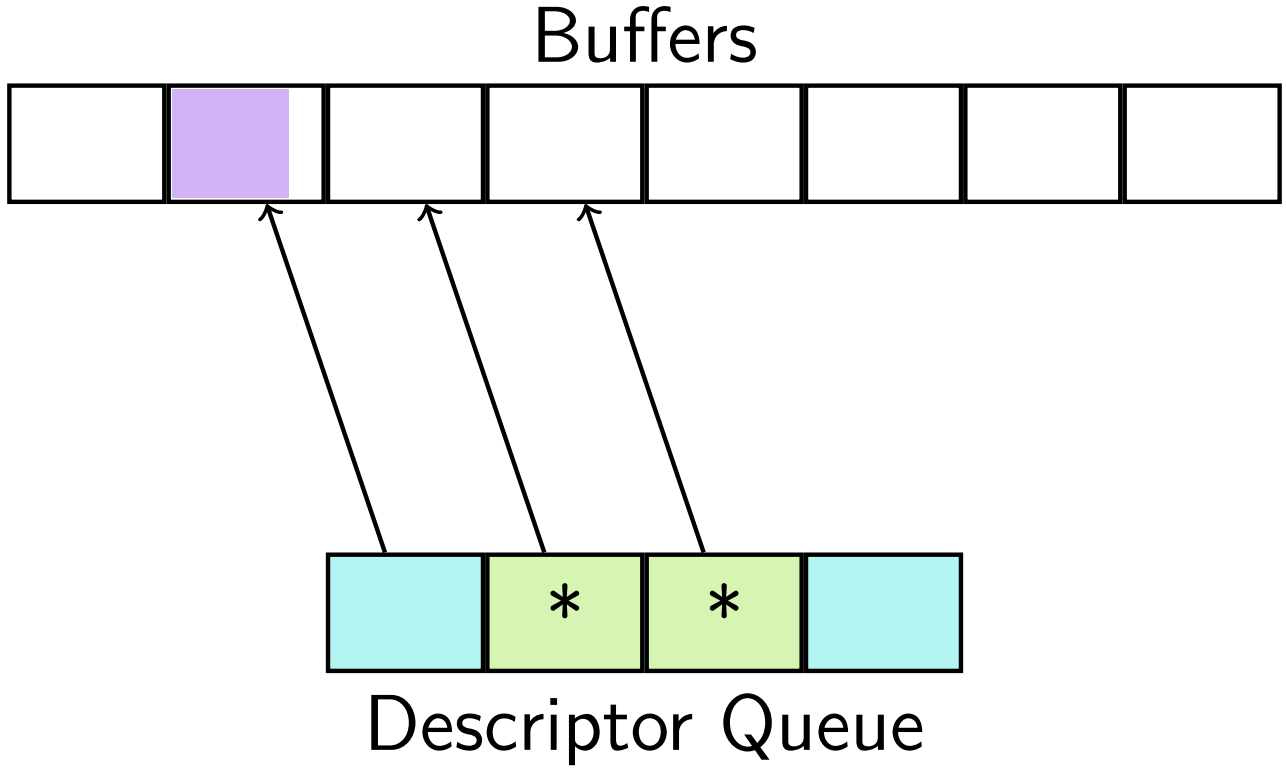
Traditional NIC DMA



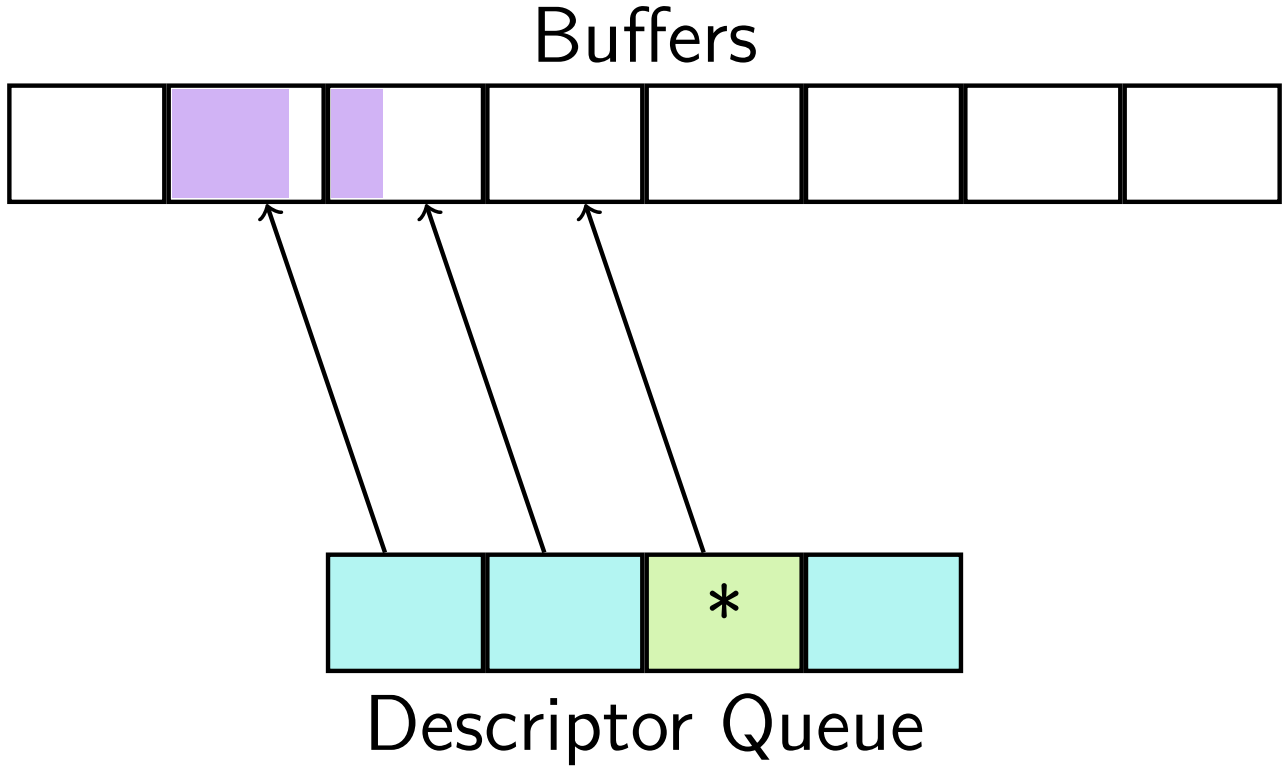
Traditional NIC DMA



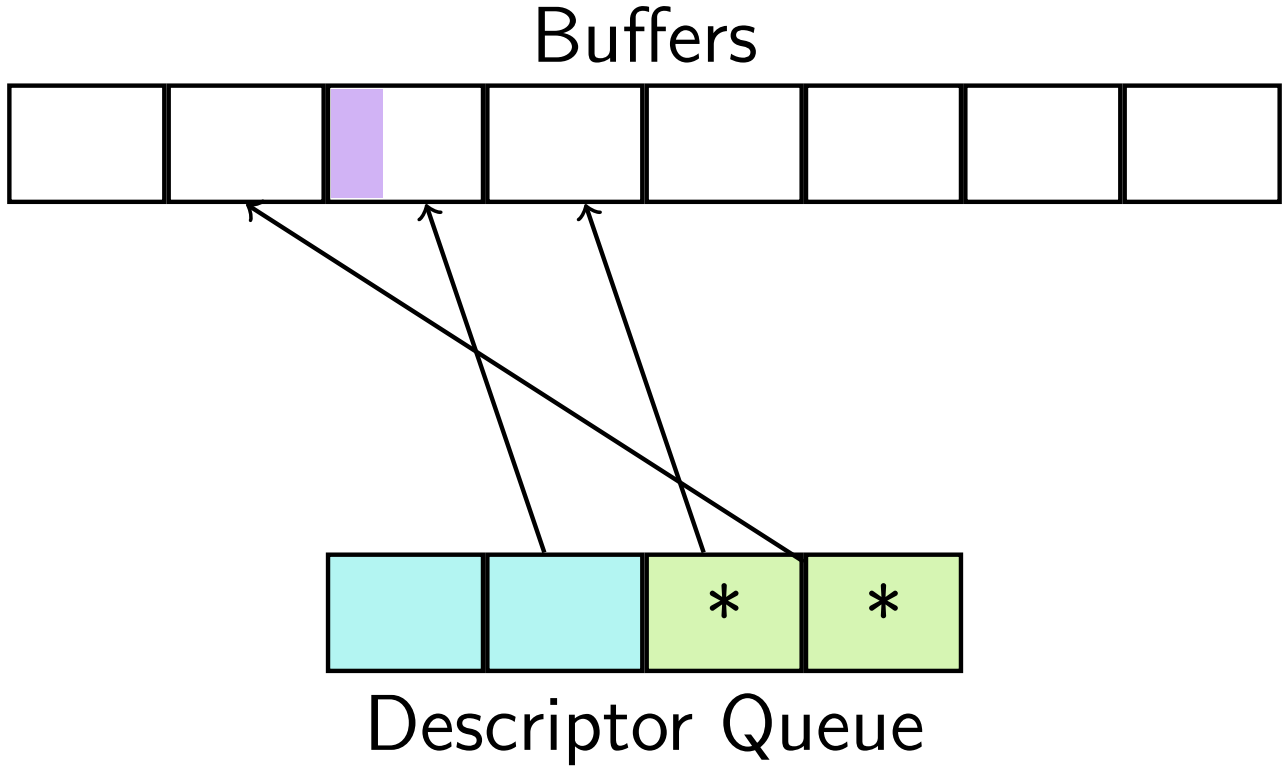
Traditional NIC DMA



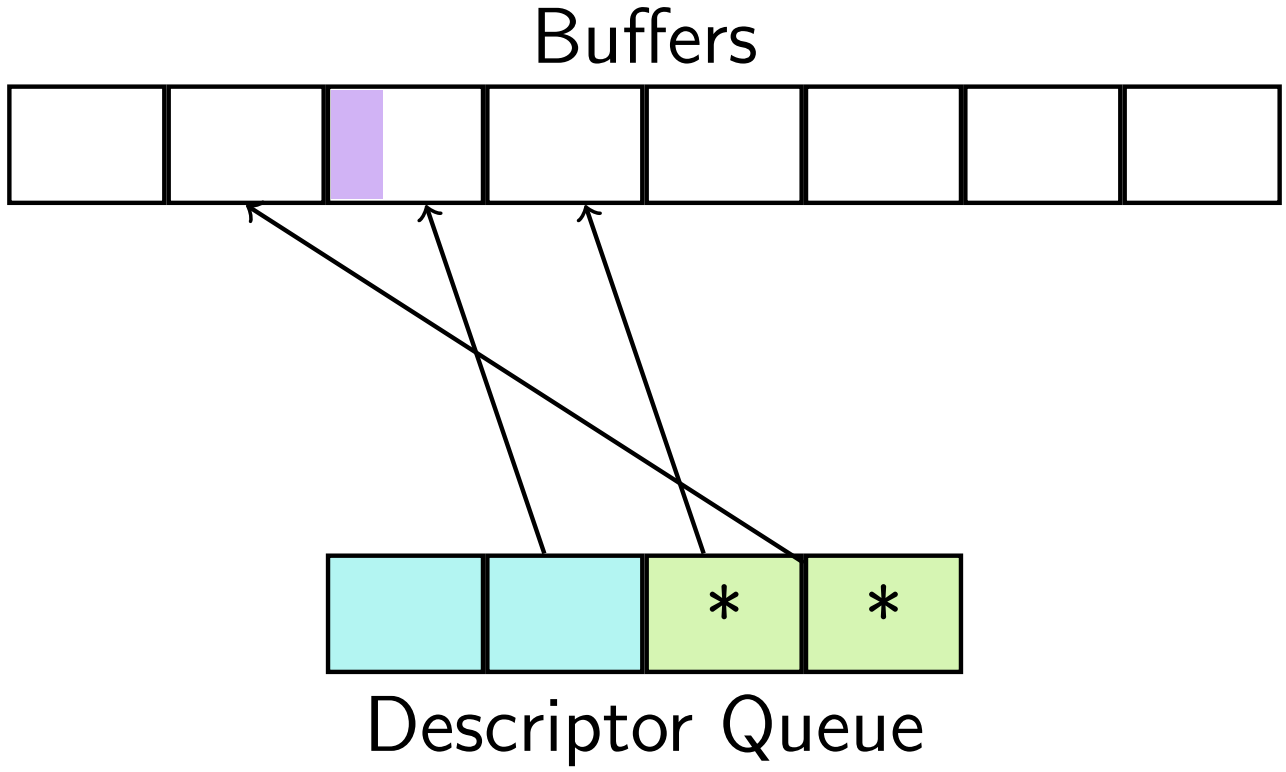
Traditional NIC DMA



Traditional NIC DMA

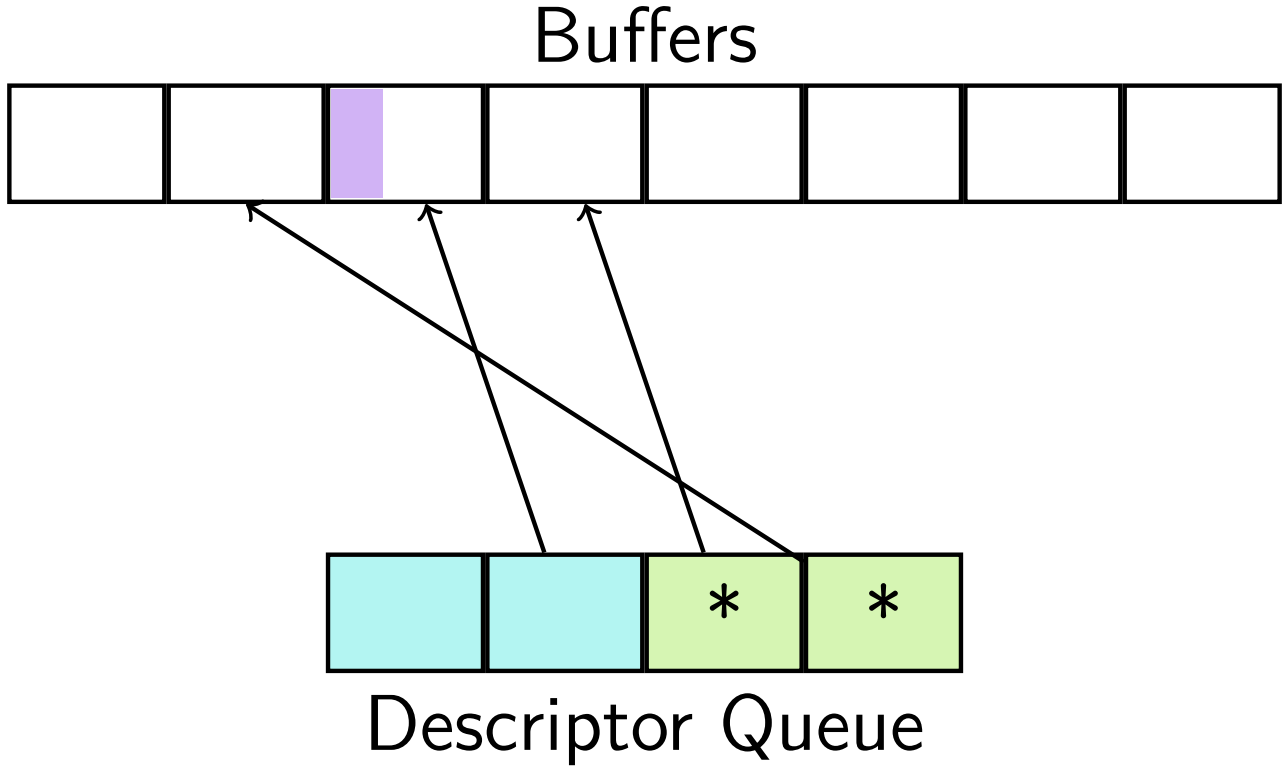


Traditional NIC DMA



- Many PCIe round-trips

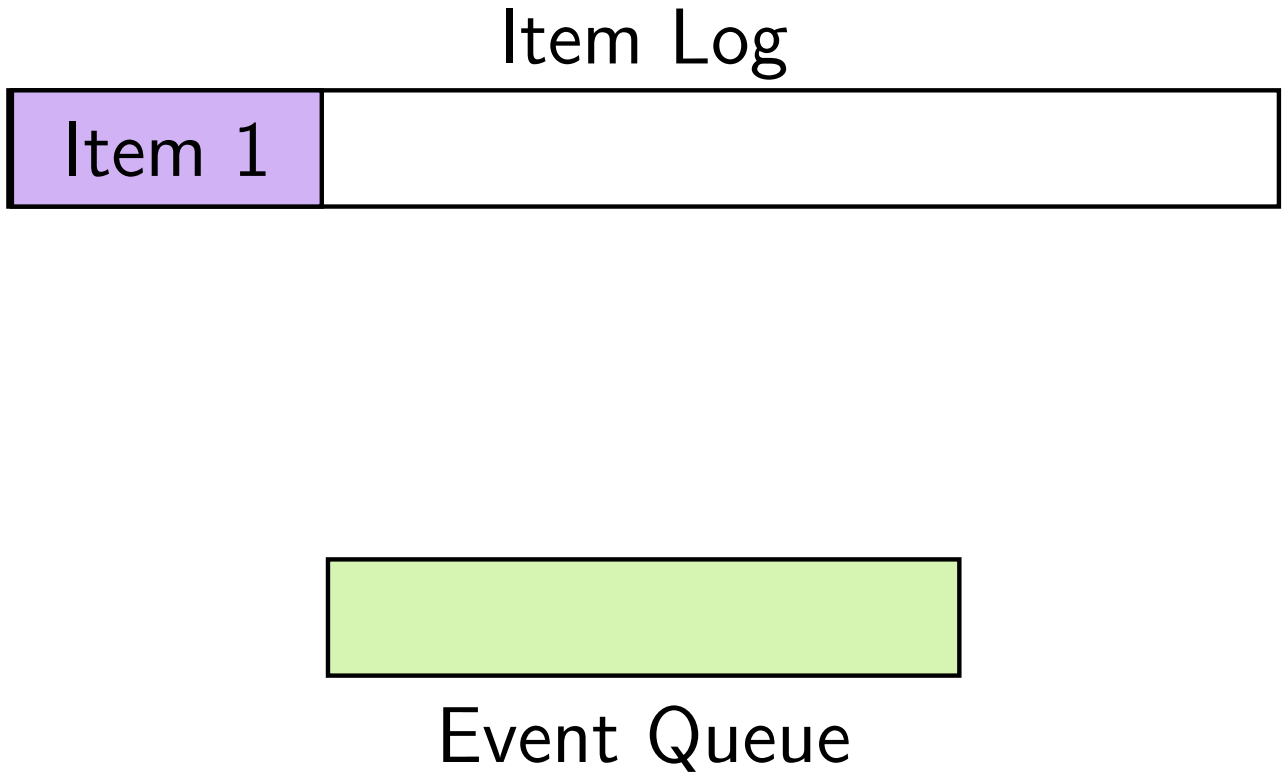
Traditional NIC DMA



- Many PCIe round-trips
- Key-Value Store: Copy items to item log

FlexNIC Key-Value Store

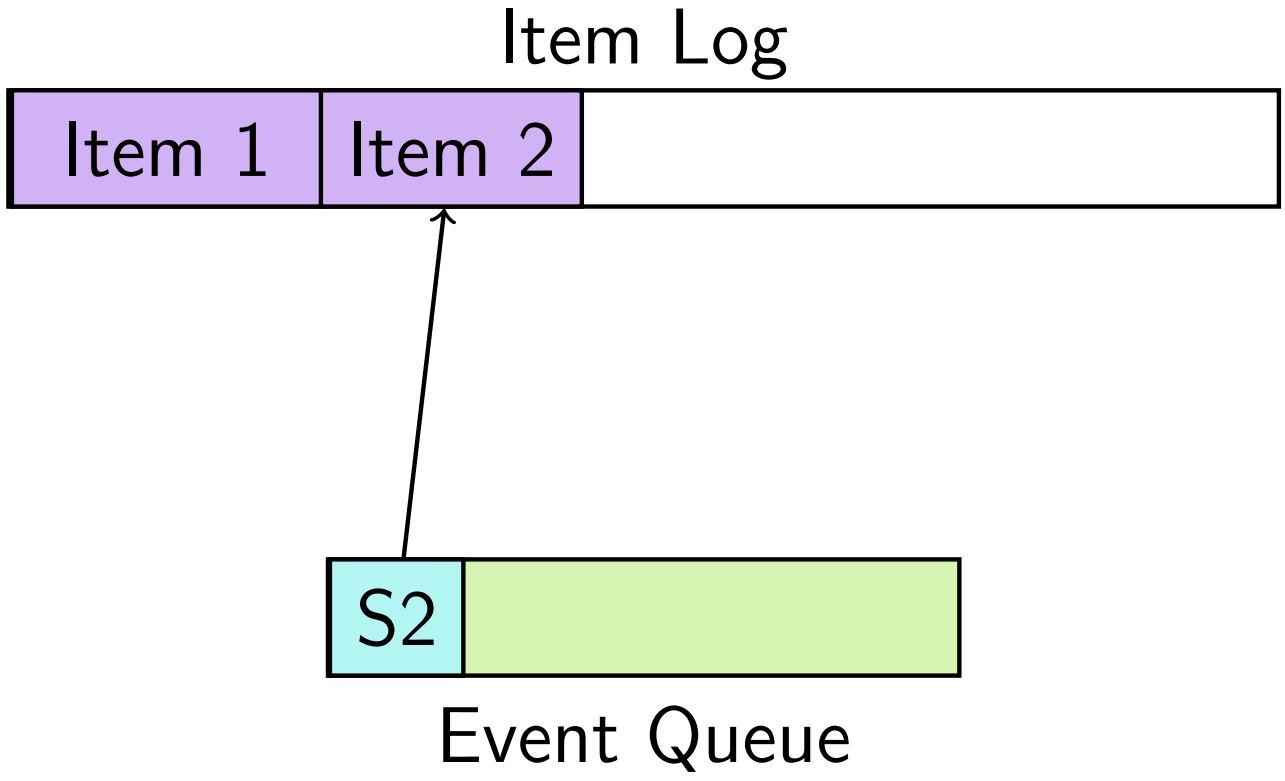
Custom DMA interface



- Combine steering, validation, and transformation

FlexNIC Key-Value Store

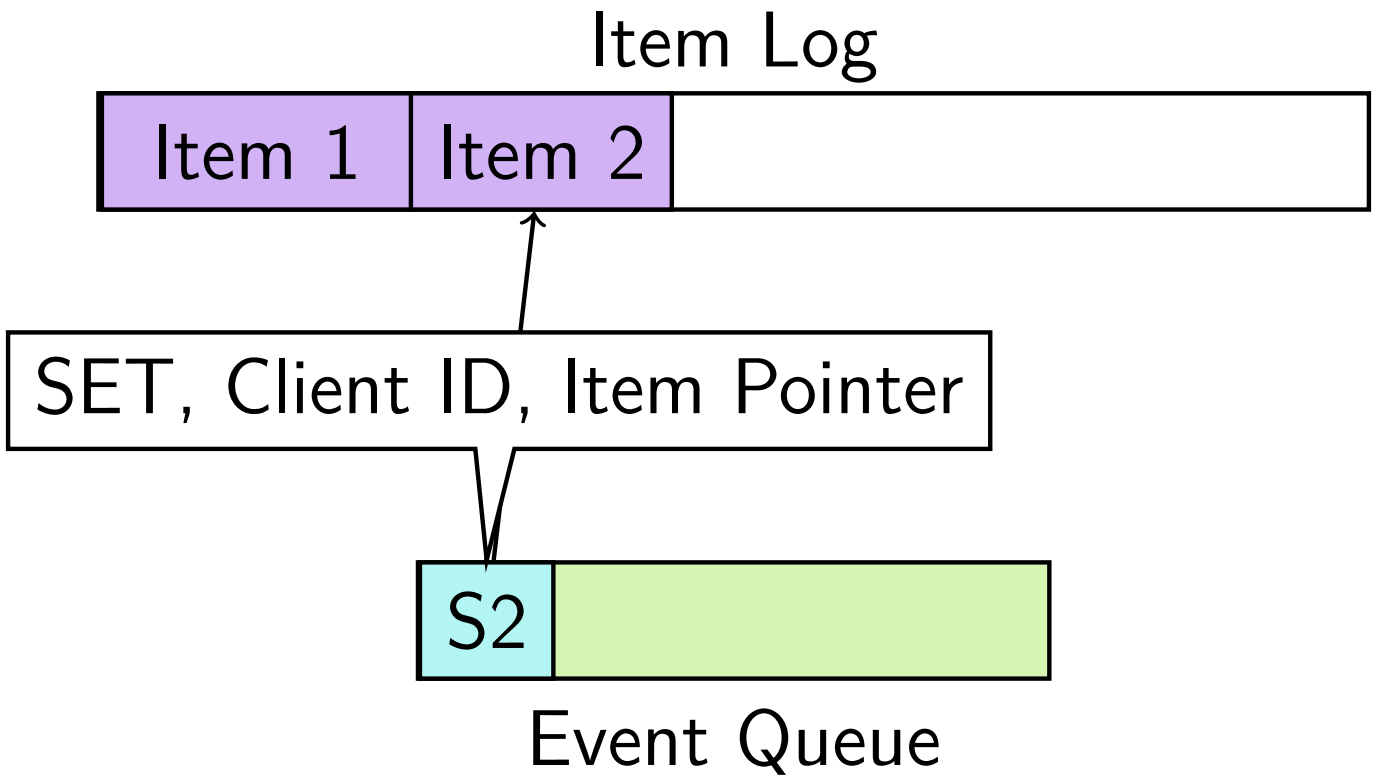
Custom DMA interface



- Combine steering, validation, and transformation

FlexNIC Key-Value Store

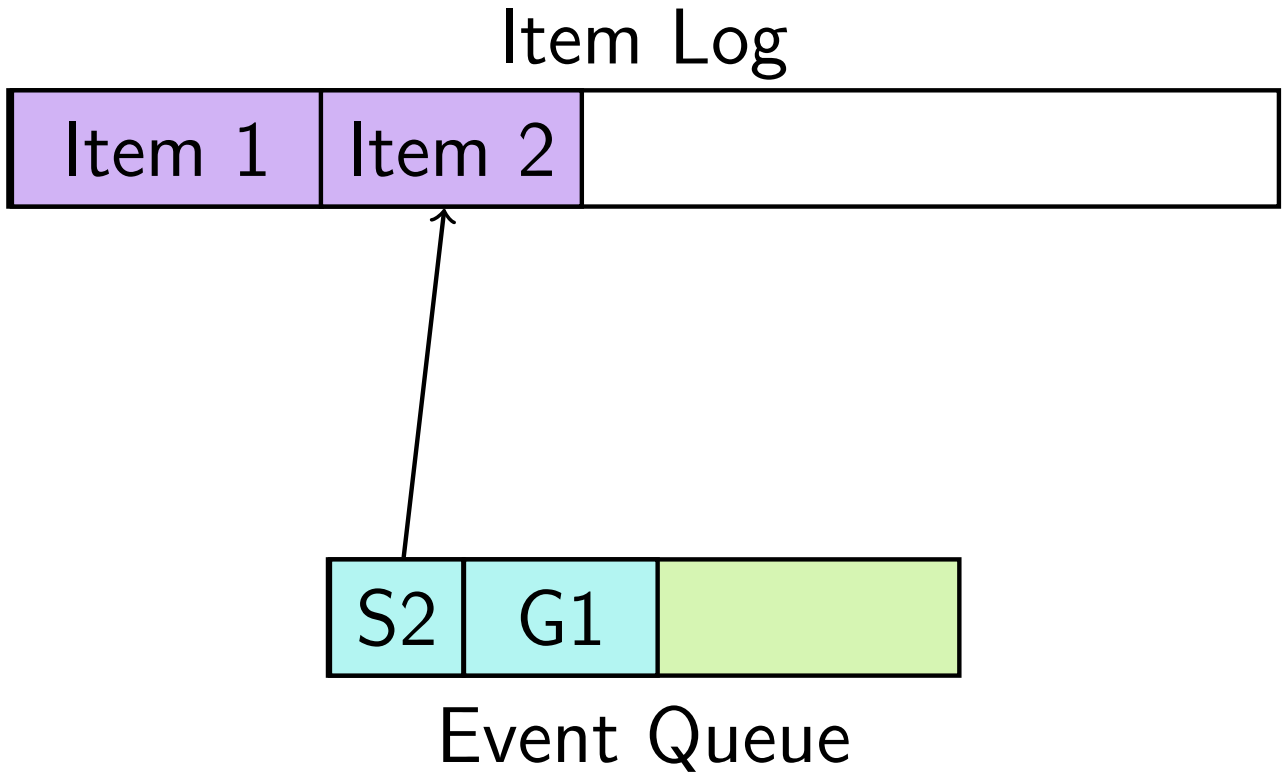
Custom DMA interface



- Combine steering, validation, and transformation

FlexNIC Key-Value Store

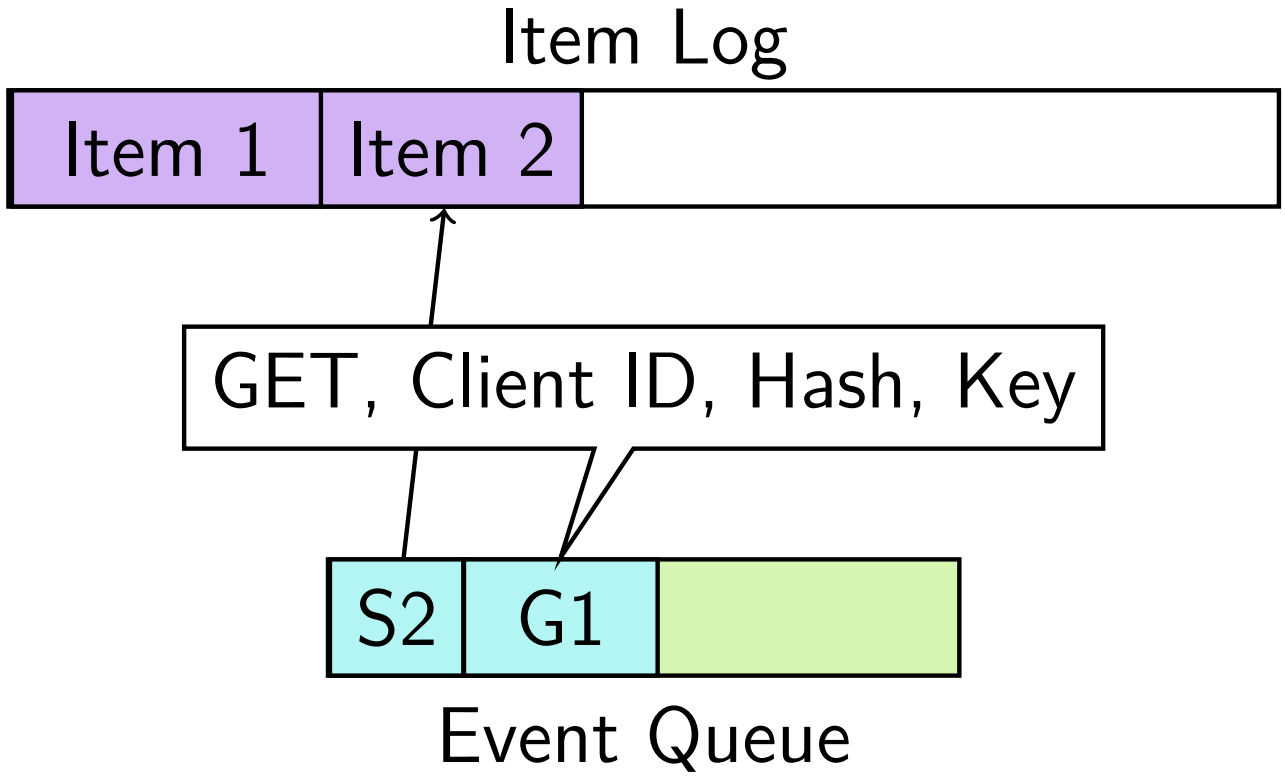
Custom DMA interface



- Combine steering, validation, and transformation

FlexNIC Key-Value Store

Custom DMA interface



- Combine steering, validation, and transformation

Preliminary Evaluation

- Lightweight key-value store implementation
- 4 core Sandy Bridge 2.2GHz
- Receive-side scaling: 1110 cycles/req
- Key-based steering: 690 cycles/req (38% speedup)
  - Emulate using RSS and IPv6 header
- FlexNIC KVS: 450 cycles/req (60% speedup)
  - Emulate in software on dedicated core

Summary

- Networks are becoming faster
  - Server applications need to keep up
- FlexNIC eliminates processing inefficiencies
  - Application control over where packets are processed
  - Efficient steering/validation/transformation on NIC
- Promising preliminary performance evaluation
  - Reduce request processing time by 60%
- Next step: evaluate more use-cases, delivery directly to cache

Backup

Further Use-case: Improving RDMA

- RDMA requires shared memory to be pinned
  - Problematic for virtualization
  - Usually mapped, but need to gracefully handle if not
  - Idea: Add rule to divert access to slow-path
- Data structure consistency with RDMA is hard
  - Often need to use messages
  - Idea: Implement data structure operations (e.g. log append, hash table insert)