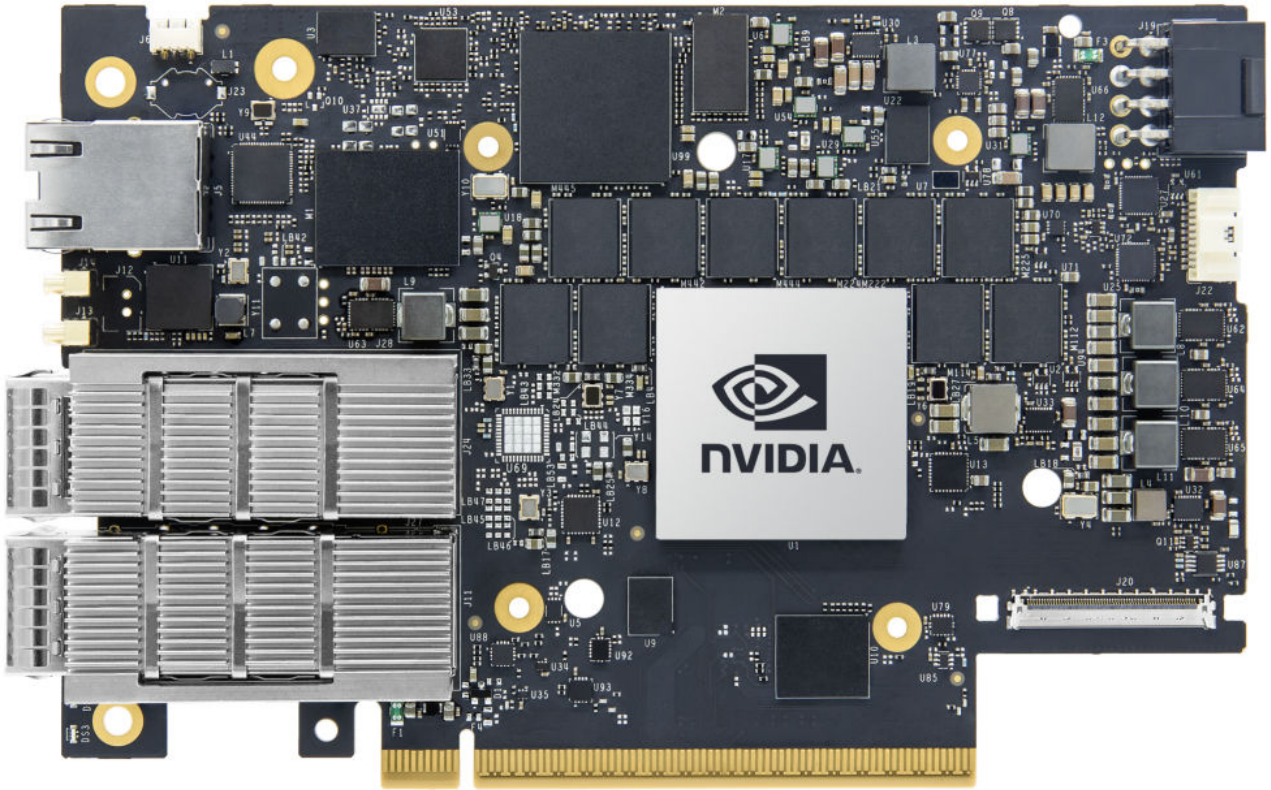


DPA-Store – An Ordered Network Data Path Key-Value Store

Frederic Schimmelpfennig[1], Jan Sass[2], Reza Salkhordeh[1],
Martin Kröning[3], Stefan Lankes[3], André Brinkmann[2]

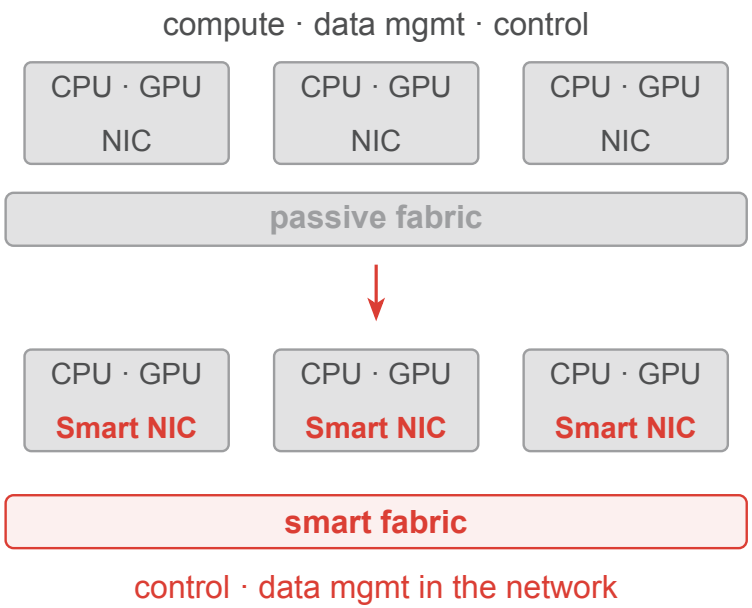
[1]JGU Mainz [2]Saarland University [3]RWTH Aachen

JG|U UNIVERSITÄT DES SAARLANDES E.ON Energy Research Center ERC RWTH AACHEN UNIVERSITY

Project Context: ScalNext

ScalNEXT

- BMBF project **ScalNext**: scalability for HPC through an **active network**.
- Today's fabric is passive – compute, data management and control all run on the hosts.
- Idea: leave computation on CPU/GPU, move **control flow** and **data management** into smart network devices.
- Devices: DPUs with on-NIC cores (BlueField), FPGA-based NICs, P4-programmable switches.
- Partners: LRZ (coordination), TU München, KIT, JGU Mainz, RWTH Aachen; runs until the end of 2025.



Bundesministerium für Forschung, Technologie und Raumfahrt
Finanziert von der Europäischen Union NextGenerationEU

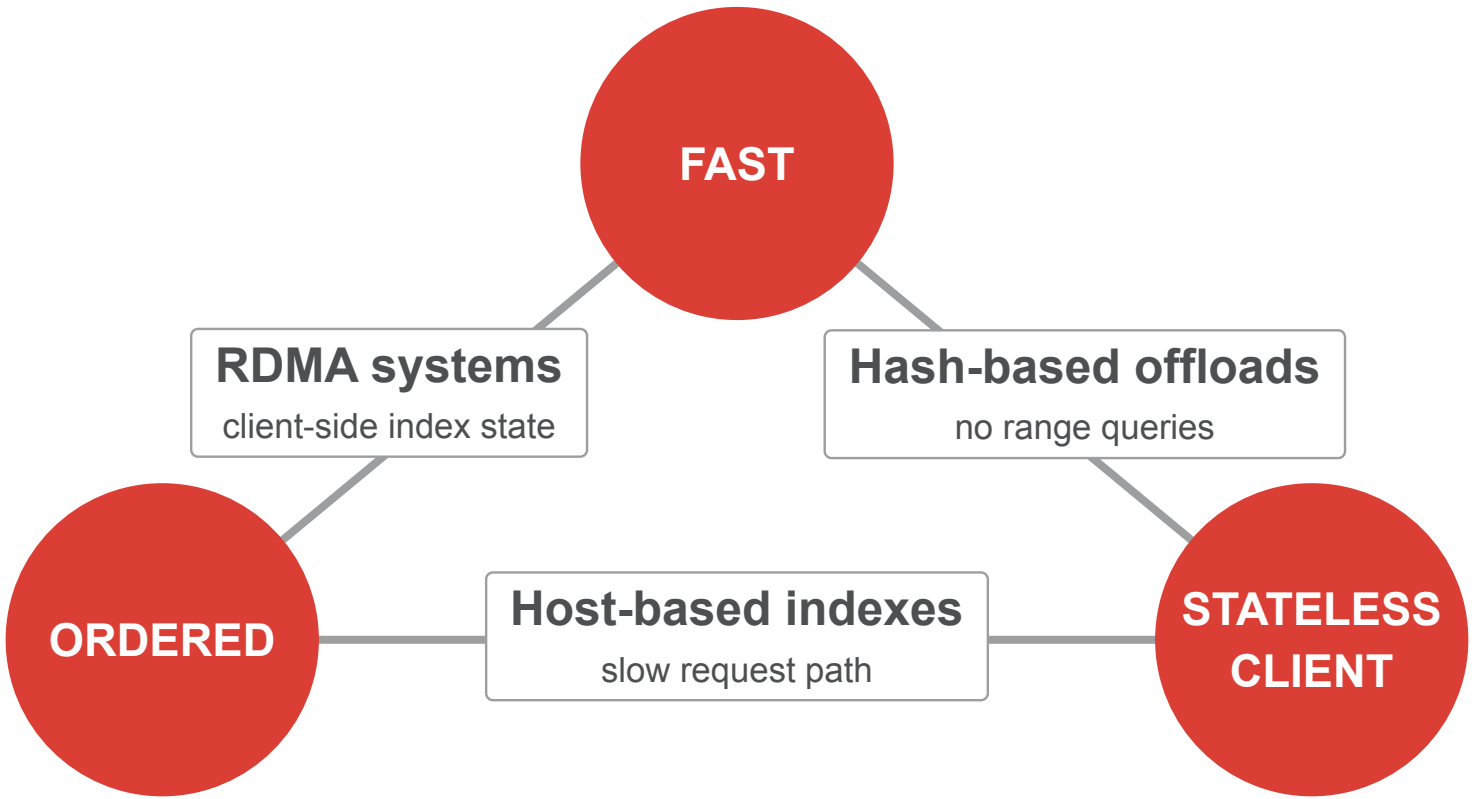
Funded by the German Federal Ministry of Research, Technology and Space and by the European Union – NextGenerationEU.

ScalNext Work Packages – and Where DPA-Store Fits

LRZ	In-network reductions on P4-programmable switches (RoCEv2); MPI progress engine offloaded to the SmartNIC.
TU München	MPI control flow on the DPU, compute kernels on host CPUs; 2D-stencil prototype shows a clear speedup.
KIT	Task-based distribution (HALAdapt) extended from node to network level; synchronisation such as barriers via SmartNICs.
JGU Mainz	Transparent file-system offload (encryption, caching) – and a KV store answered by the NIC itself: **DPA-Store, this talk**.
RWTH Aachen	OS management tasks offloaded to the SmartNIC; OS bypassing.

Project-level lesson: offloading *all* control to today's DPUs is not realistic – what is missing is more direct access from the DPU to host memory. Offloading individual tasks already pays off.
DPA-Store shows both sides: 33 MOPS on the GET path, and the host→DPA write wall at 1.7 MOPS INSERTs.

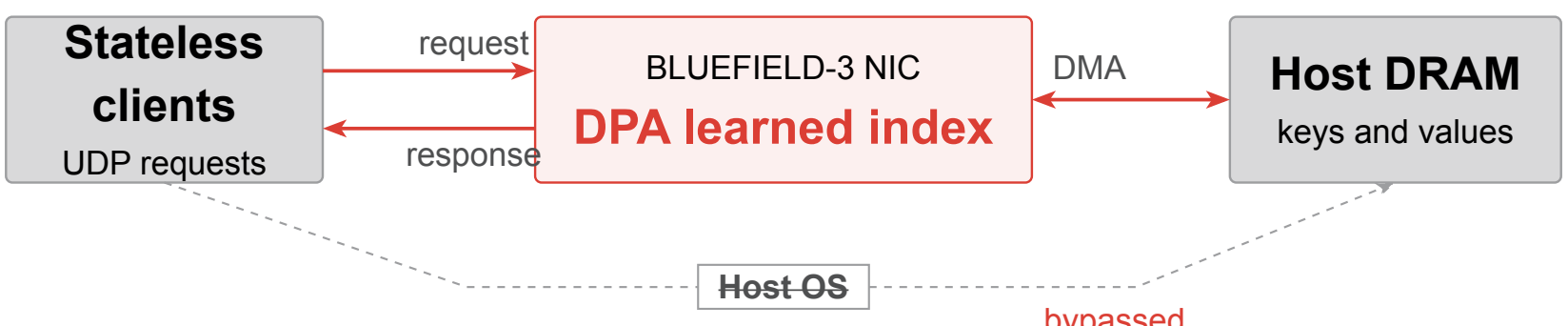
Remote KV Stores Force a Tradeoff



KV-Store at Data Path Level: DPA-Store

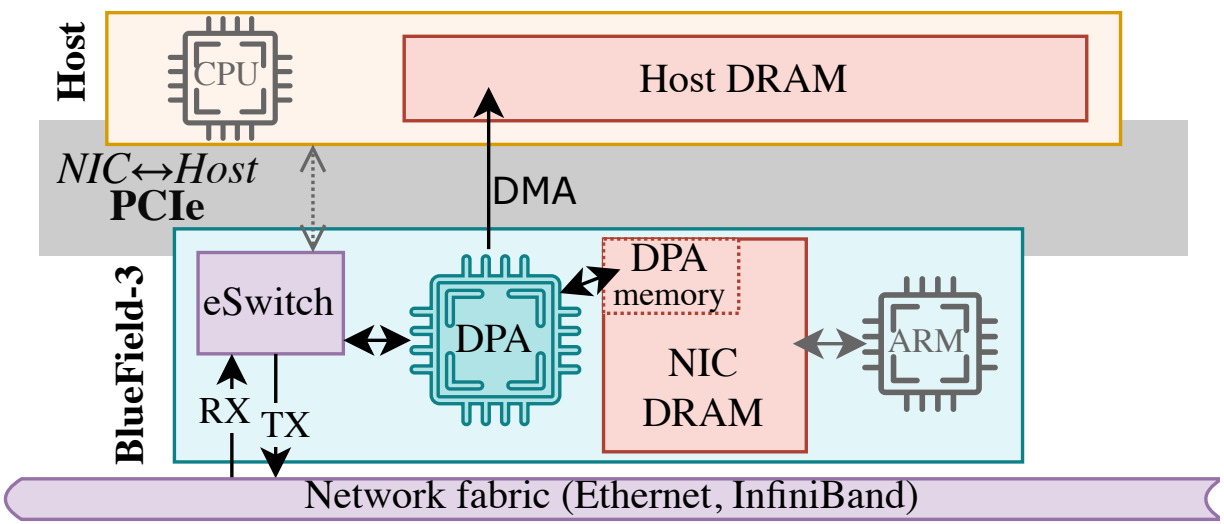
- Ordered in-memory KV store on the NVIDIA BlueField-3 SmartNIC.
- Requests terminated **on-path** by Data Path Accelerators (DPAs). No OS involved.
- Learned index on the NIC; values in host memory.

33 MOPS GET
13 MOPS RANGE



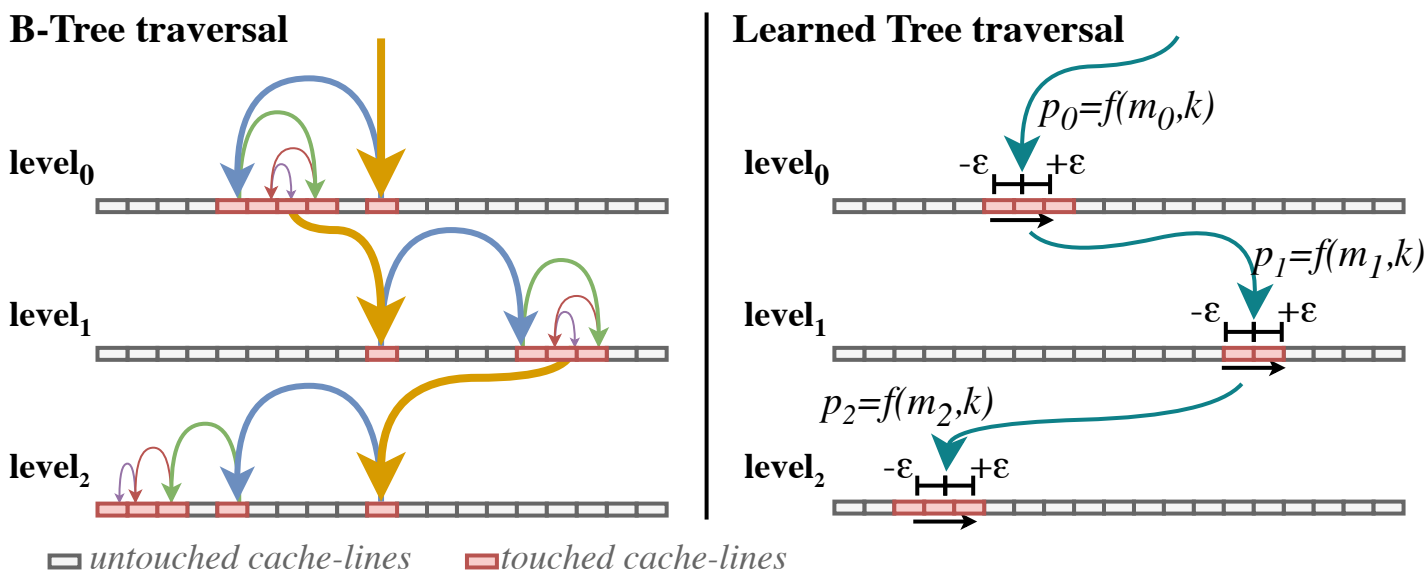
Background: BlueField-3 DPA Subsystem

- 16 RISC-V cores $\times$ 16 threads $=$ 256 threads (189 usable), hardware-scheduled.
- Great parallelism, down to memory-access granularity.
- **On-path**: packet matching invokes DPA code. No OS.
- DMA access to host memory.
- The catch. *Not* a small CPU:
  - **465 ns** memory access ($\sim 5\times$ host DRAM)
  - **1 GiB** DPA memory
  - **no floating point**

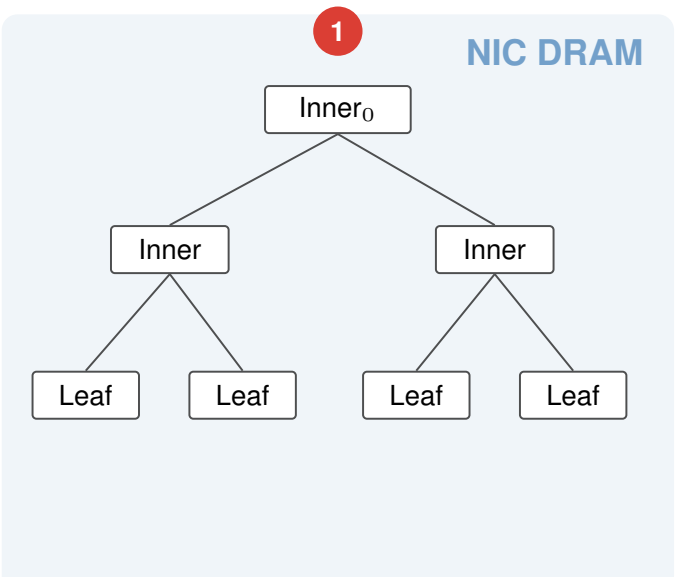


Cache-Line Access Efficiency

- **Goal: minimize cache-line accesses.**
- B$^+$-tree: scattered cache-line accesses on every level.
- DPA-Store uses a learned index (PLA): predict $p = a \cdot k + b$, scan only $[p - \epsilon,\ p + \epsilon]$.
- We chose small ϵ: inner $\leq$ 2 cache-lines, leaf $\leq$ 3 DMA cache-lines.

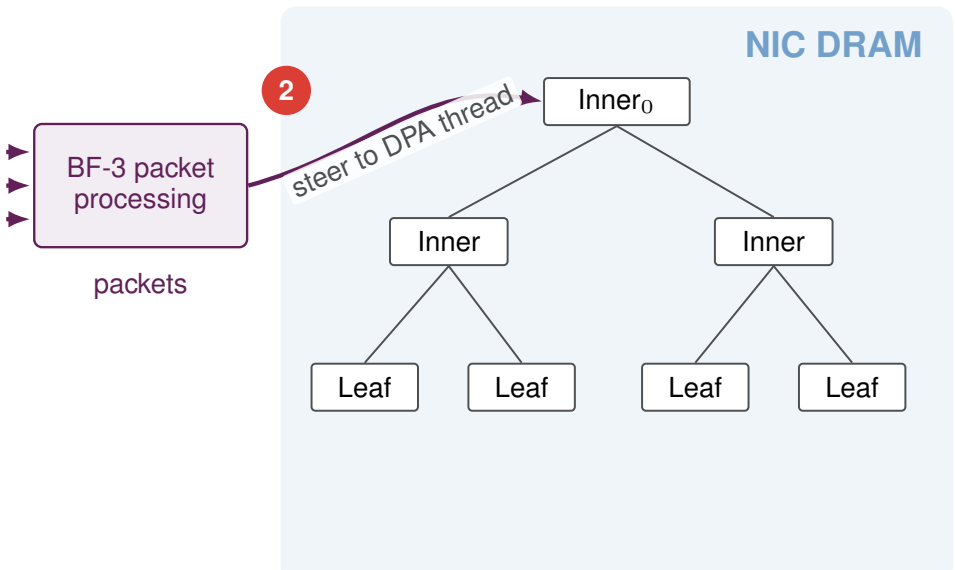


Architecture Overview



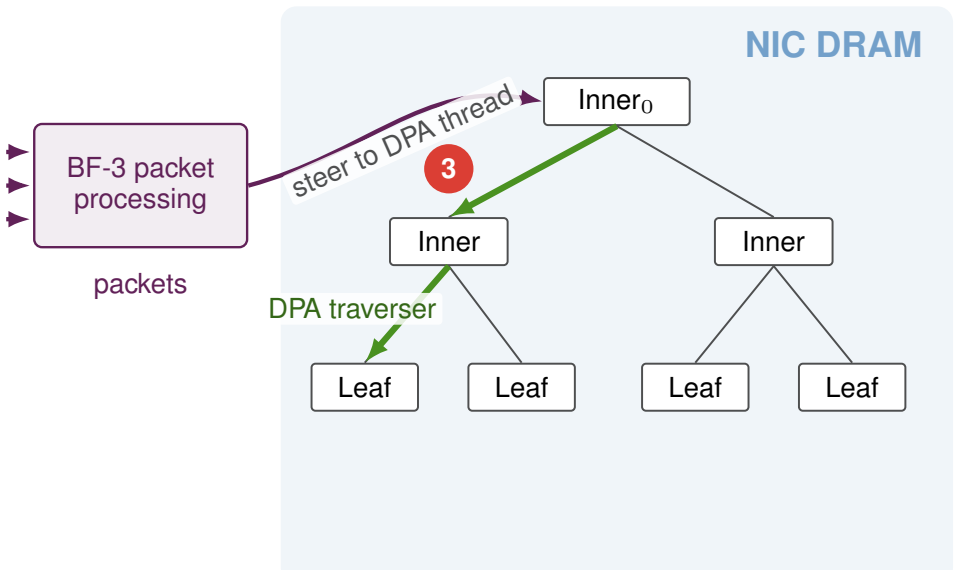
1: The learned index resides in NIC DRAM.

Architecture Overview



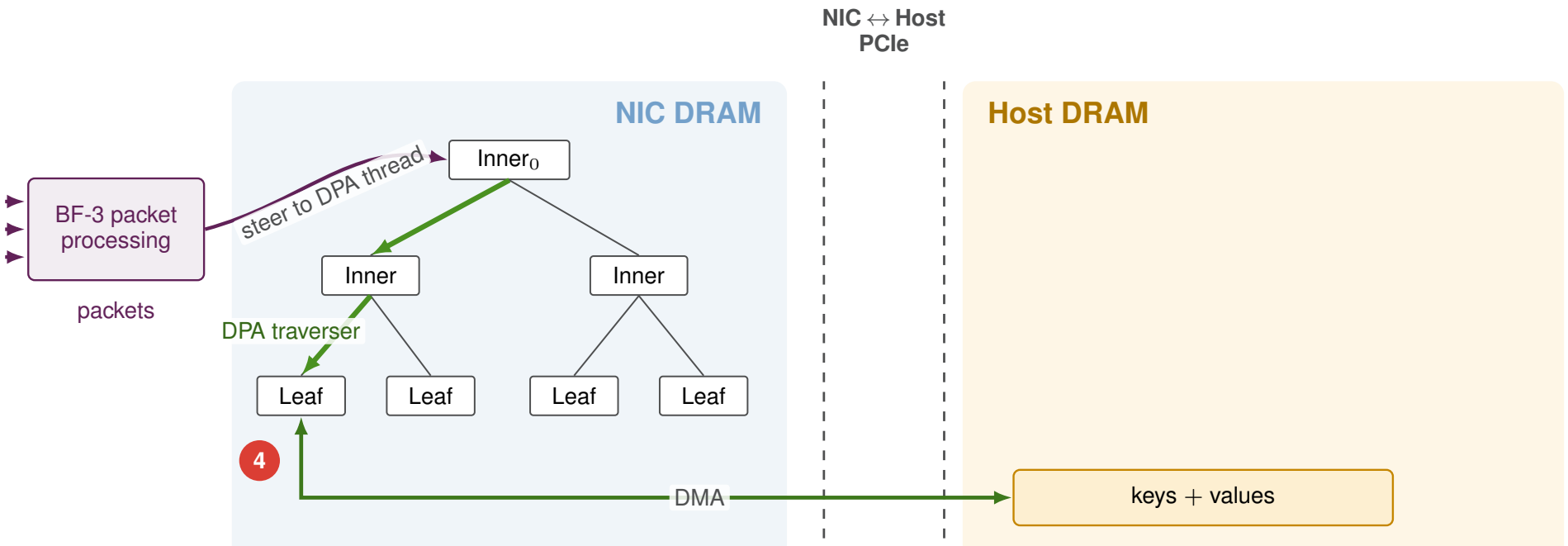
2: BlueField-3 steers packets directly to DPA threads.

Architecture Overview



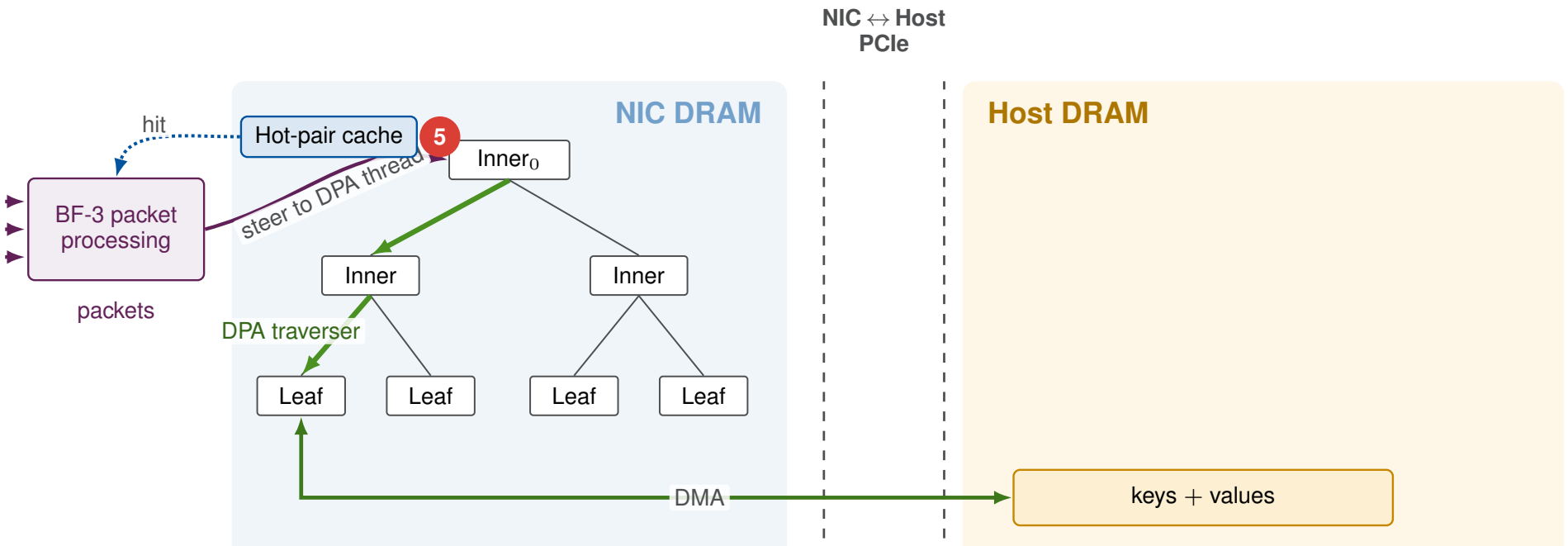
3: DPA traversers walk the learned index on the request path.

Architecture Overview



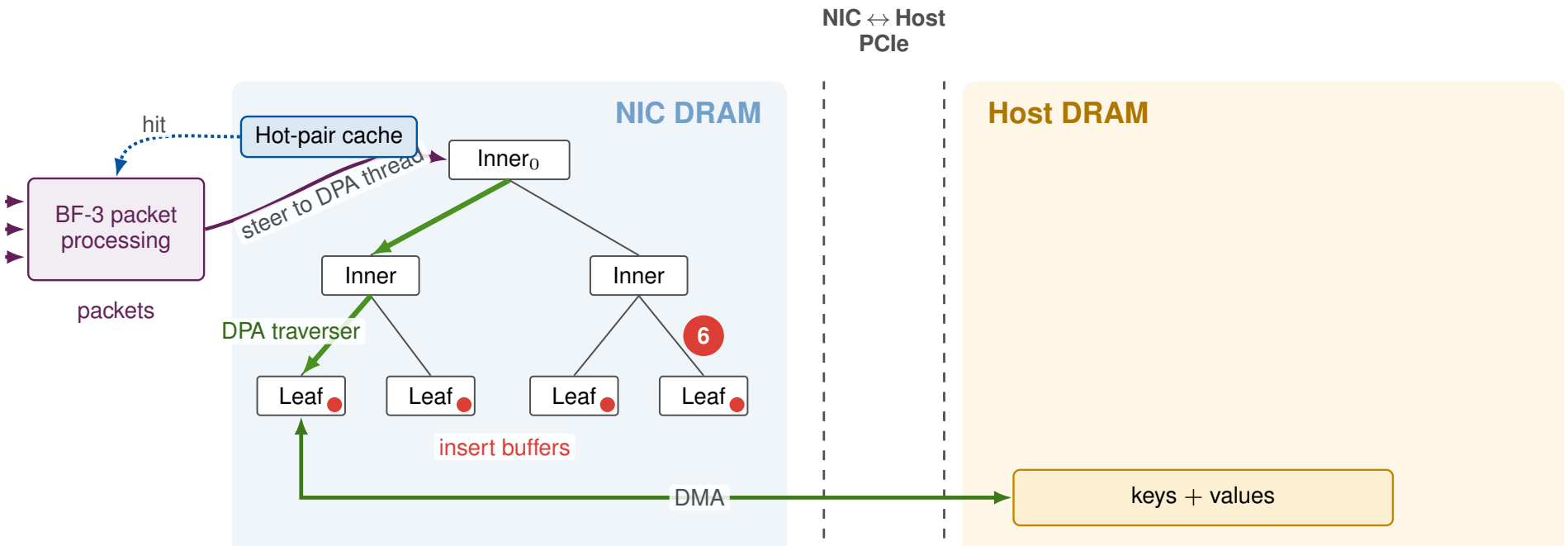
4: DPA traversers fetch values from host memory via DMA.

Architecture Overview



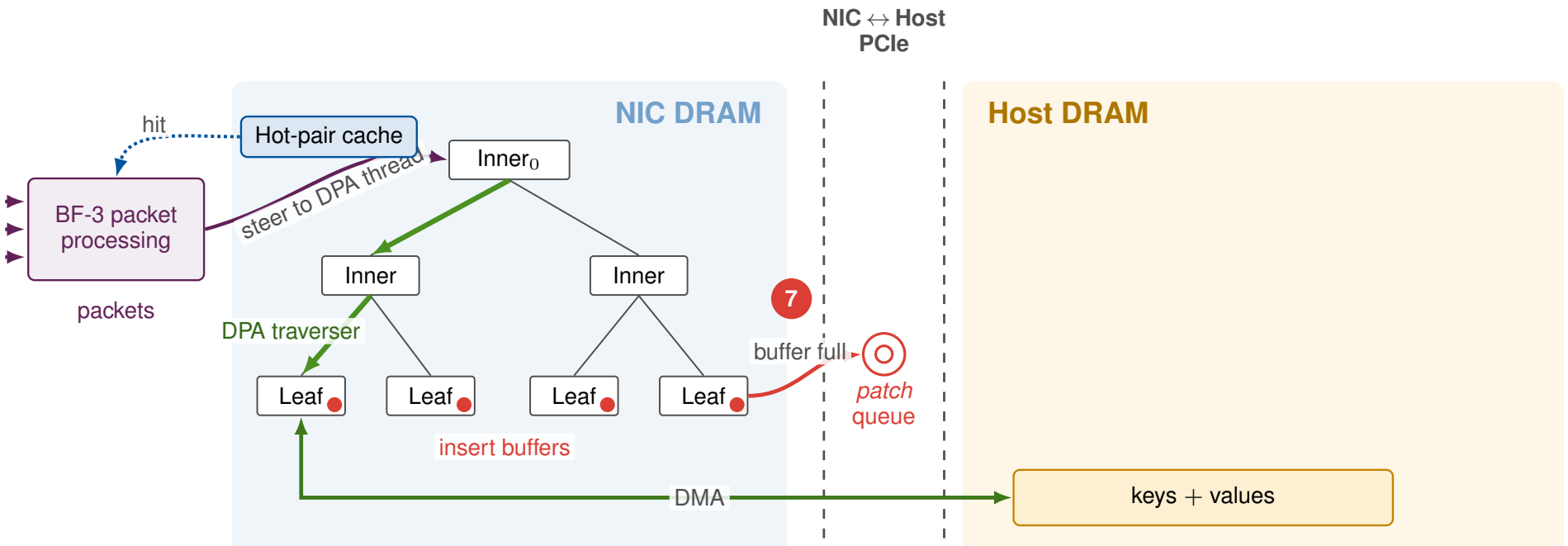
5: A NIC-side cache serves hot key-value pairs immediately.

Architecture Overview



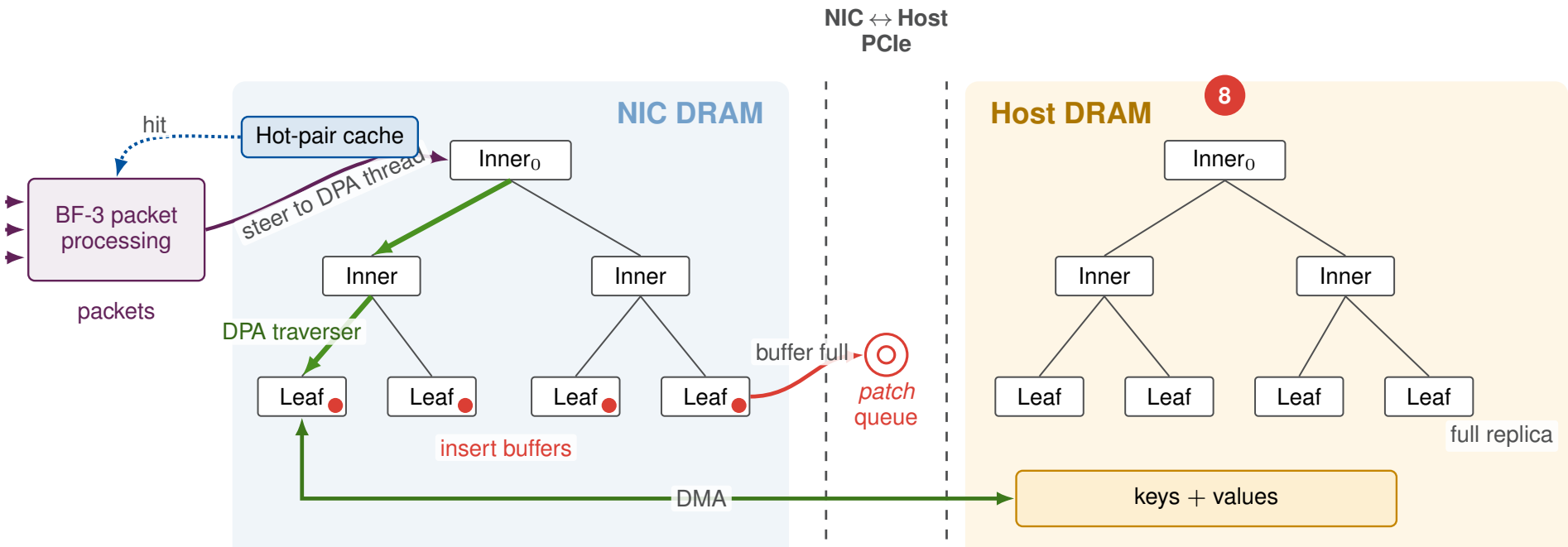
6: Leaf insert buffers make writes immediately visible.

Architecture Overview



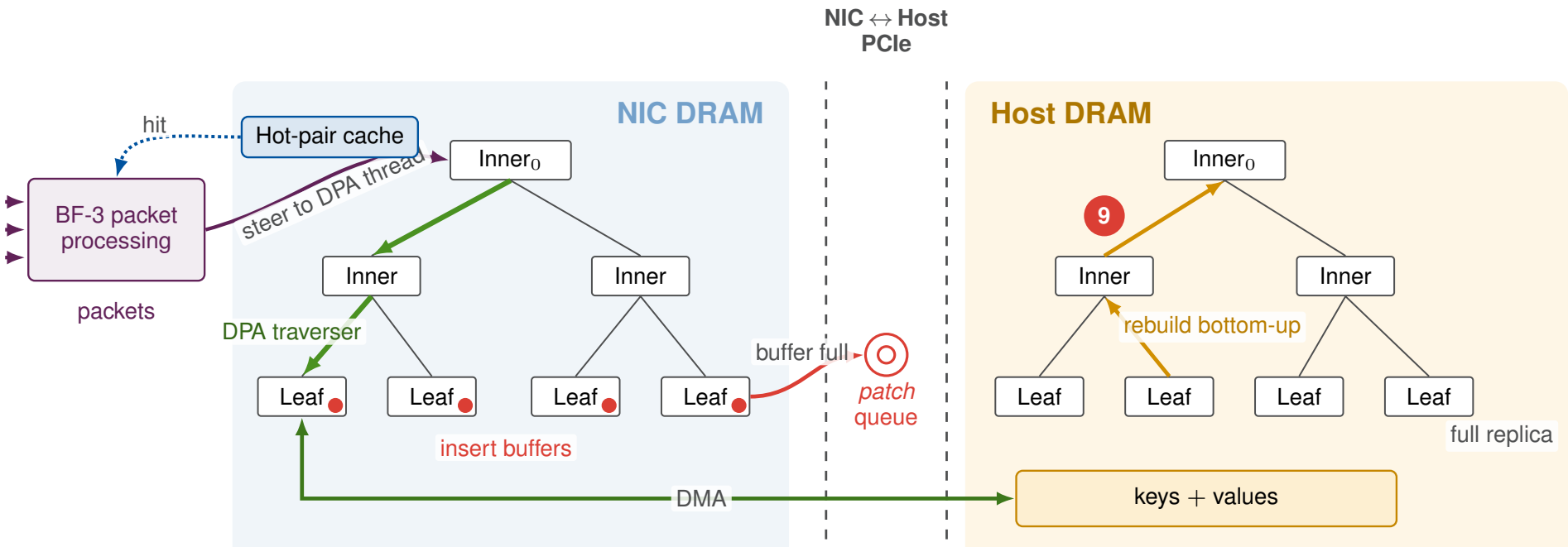
7: A full insert buffer triggers a patch request.

Architecture Overview



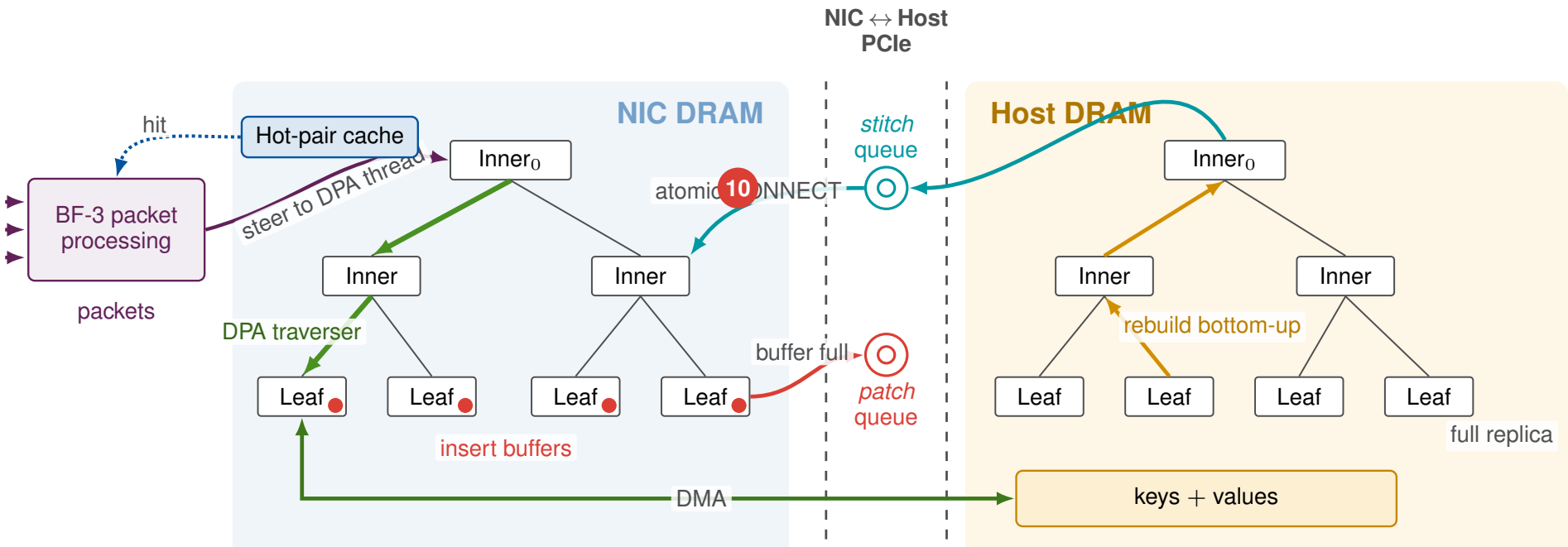
8: The host maintains a full replica of the tree.

Architecture Overview



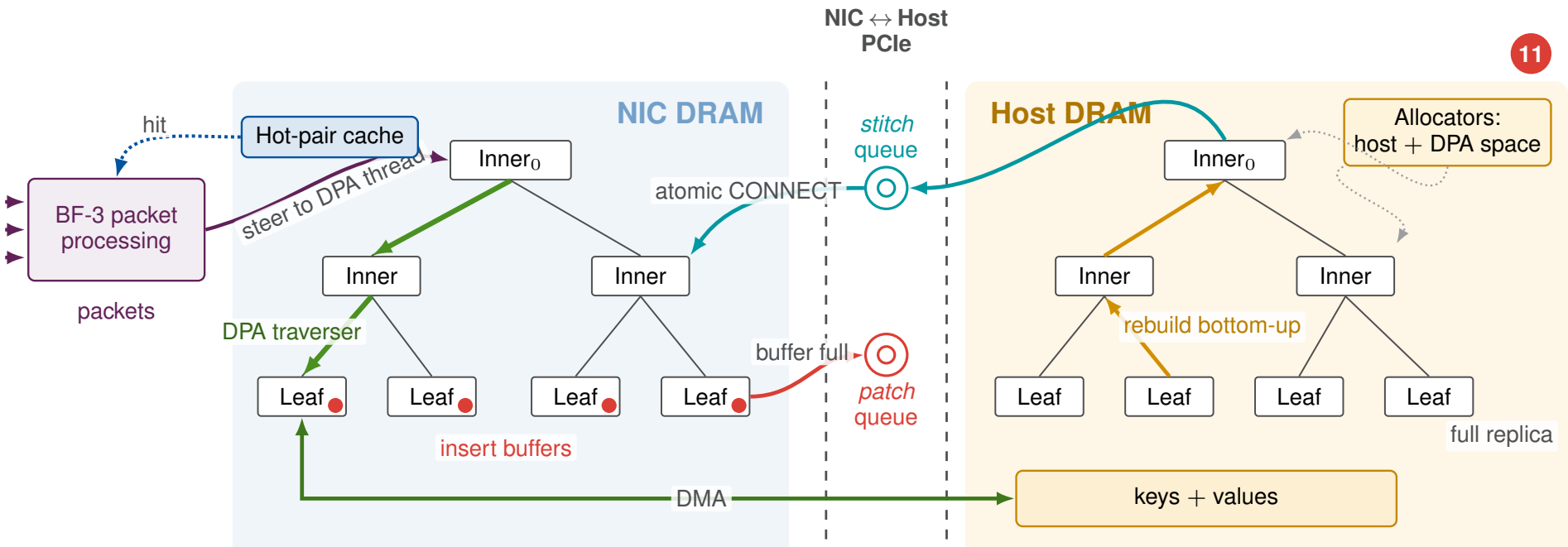
9: Host patchers rebuild affected nodes bottom-up.

Architecture Overview



10: Stitchers apply rebuilt nodes to the NIC-side tree.

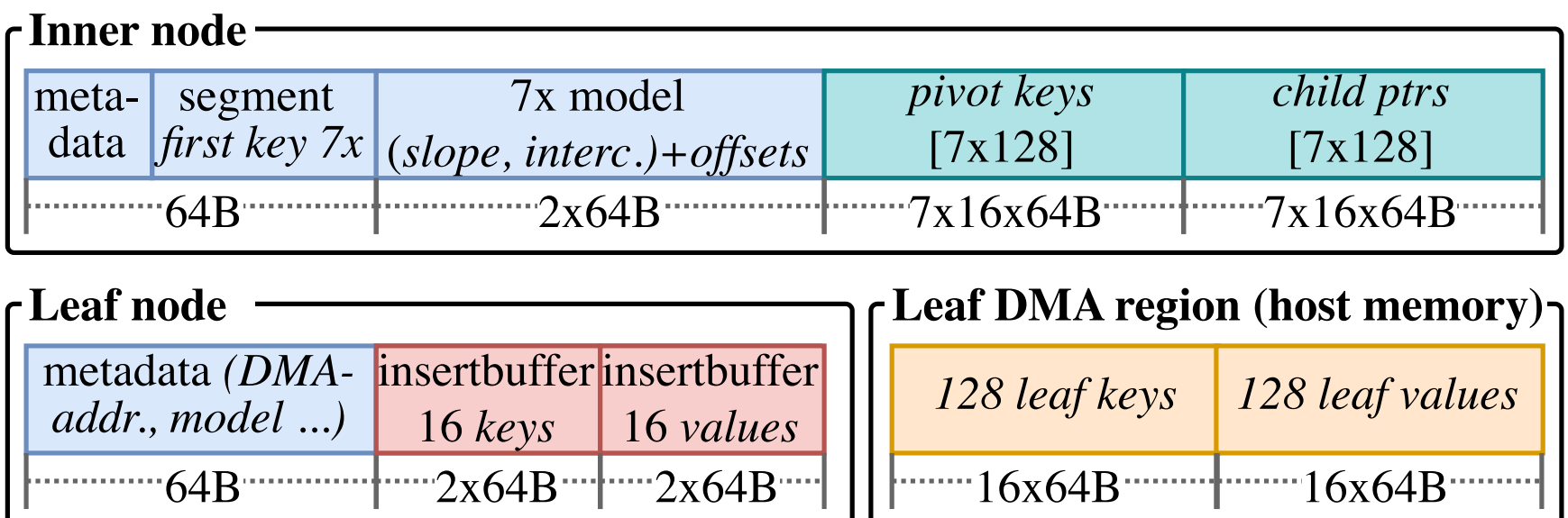
Architecture Overview



11: Separate allocators manage host and DPA memory.

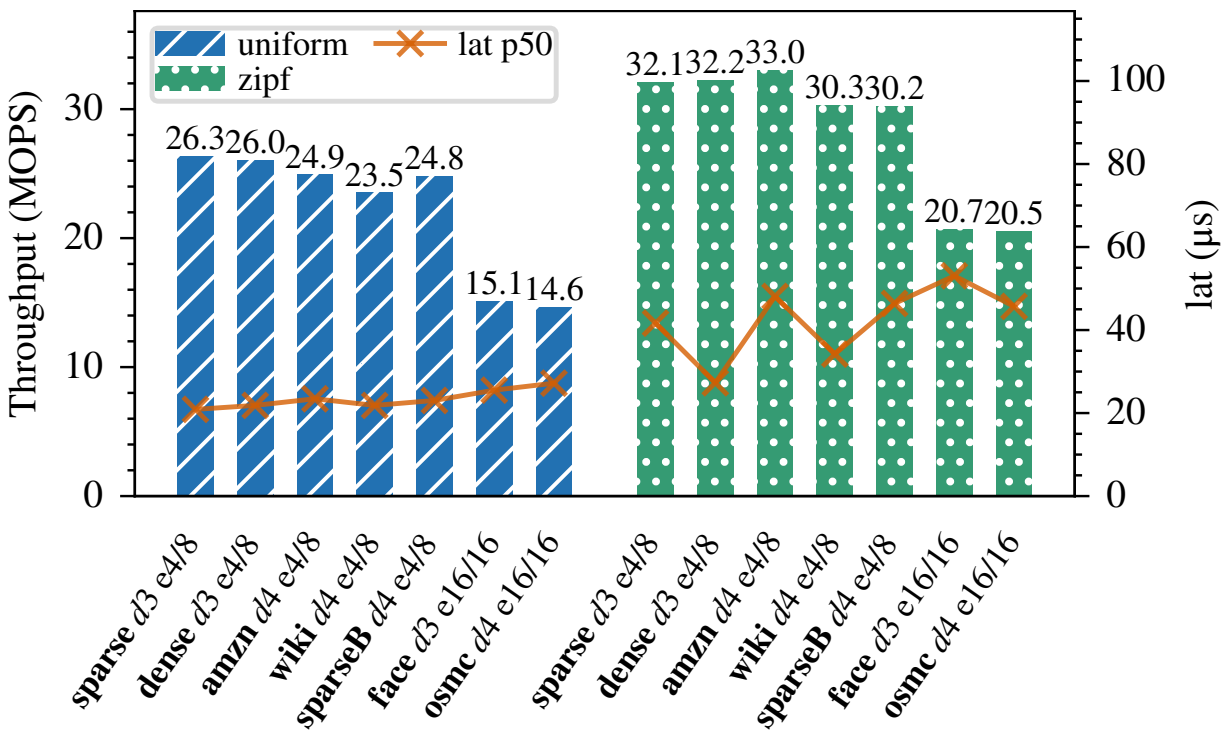
Memory Layout and Prefetch Optimization

→ Memory accesses are prefetched to allow maximum concurrency with the DPA hardware scheduler.



Evaluation: GET and the Memory Model

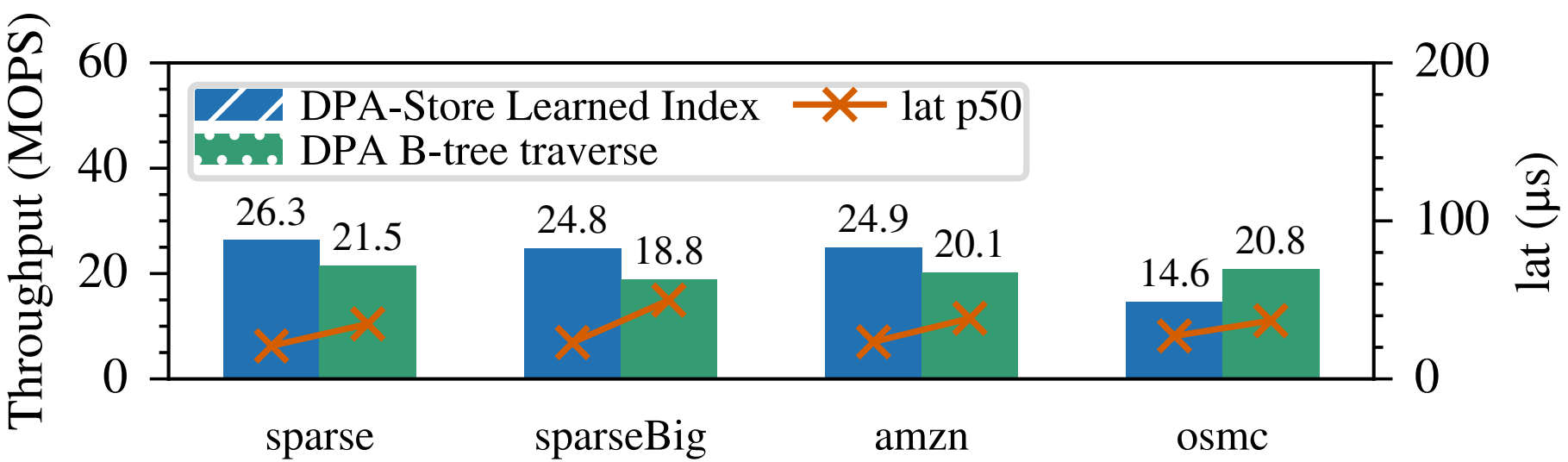
- B3140L, 100 GbE, 6 DPDK clients; SOSD data, 25M keys.
- GET up to **33 MOPS**, ∼26 without hot cache; RANGE **13 MOPS**.
- Cache-line model predicts **27–31 MOPS**. Matches.
- → Hitting the DPA **memory latency wall**.



Was the Learned Index Worth It?

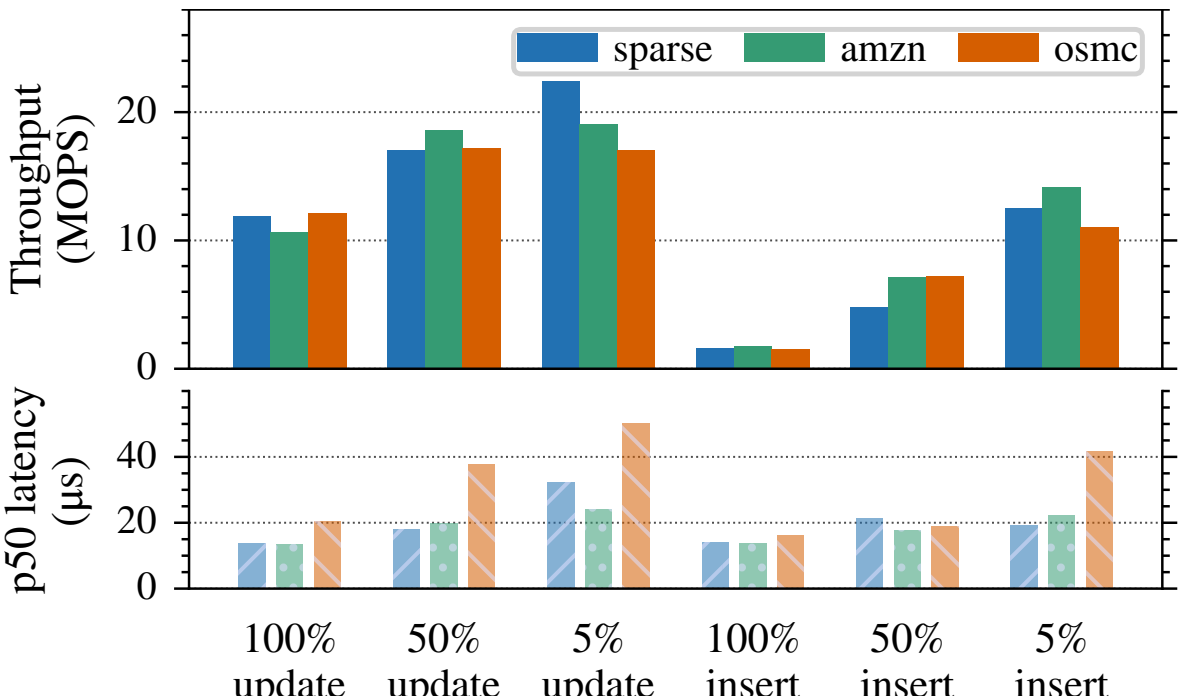
- Same system with a B$^+$-tree instead: learned index wins on most datasets.
- Governed by ϵ: small $\rightarrow$ contiguous windows win. Larger $\epsilon \rightarrow$ B$^+$-tree competitive.
- Middle path: B$^+$-tree with learned interpolation, throughput in between.

`osmc` runs $\epsilon = 16$ (default 4/8)
to limit index overhead



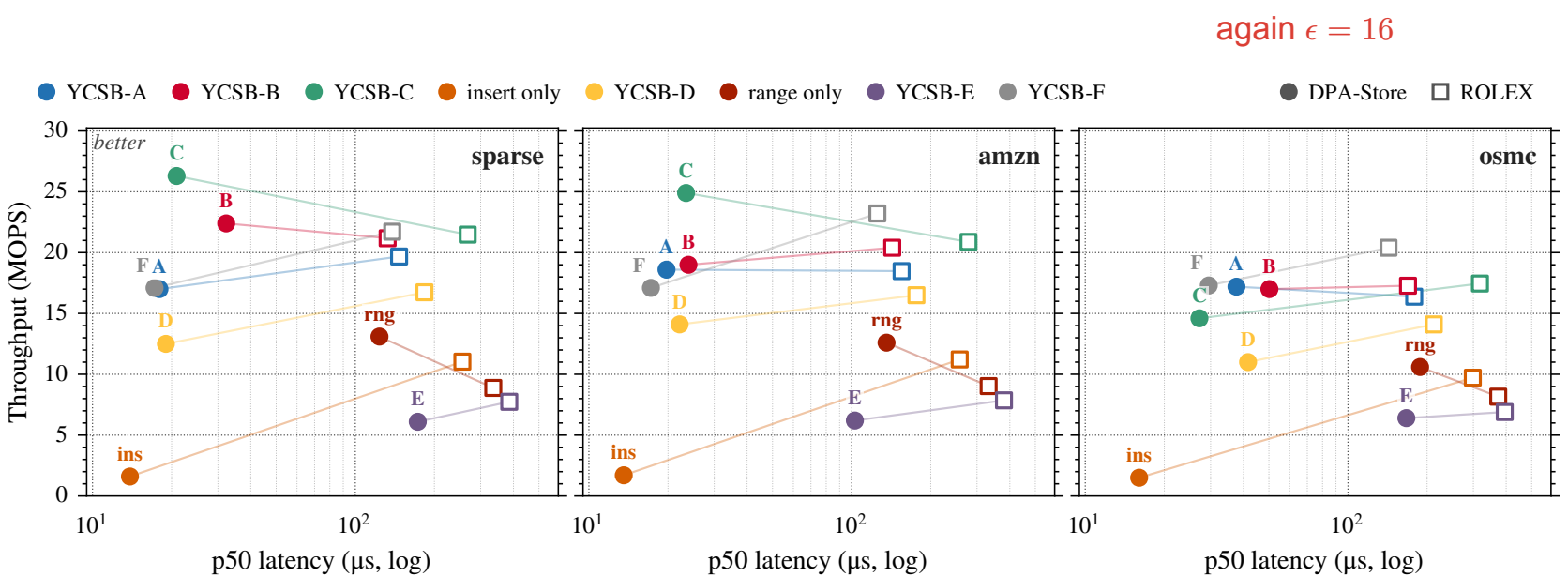
INSERT at 1.7 MOPS

- UPDATEs are fine: 12.1 MOPS.
- INSERTs cap at **1.7 MOPS**: **bottlenecked host→DPA write path**.
  - Likely a BF-3 limit accessing DPA memory fine-grained.
- Holds for both: DMA pull and explicit NVIDIA FlexIO push.
- Mixed workloads stay fast: no cross-contamination.



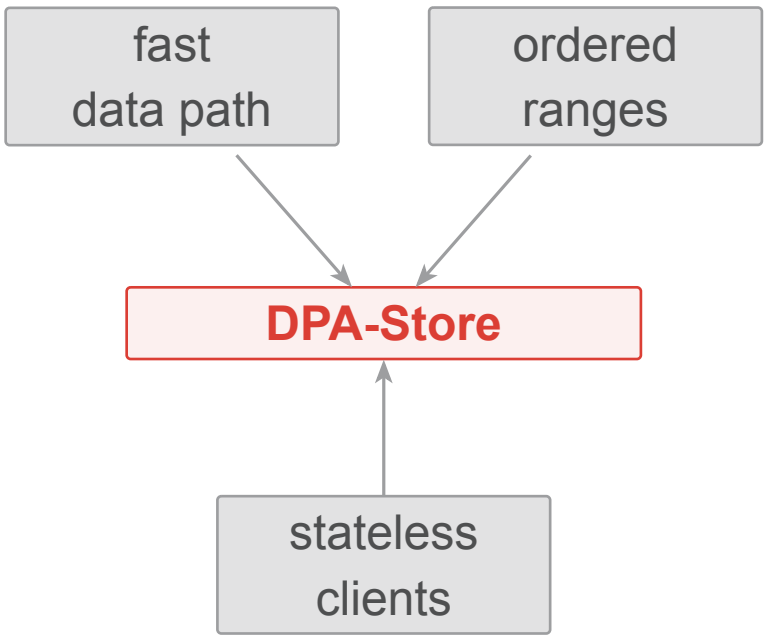
Compared with ROLEX: RDMA Flagship

- ROLEX = state-of-the-art RDMA learned-index KV store. Same hardware and testbed.
- DPA-Store wins YCSB-A on `amzn`/`osmc`, RANGE, GET on `sparse`/`amzn`. Lower latency.
- ROLEX wins insert-heavy mixes: one-sided RDMA writes, clients keep index state.
- Same ballpark, but DPA-Store clients hold **zero state**.



Takeaways

- Ordered KV, high performance, and stateless clients are possible on the NIC data path.
- Recipe: maximum concurrency · lock-free operation · cache and memory optimization · OS bypass for traversals · maintenance on the host.
- Small ϵ worked well for most datasets we tested.
- BF-3 walls: memory latency + host→DPA writes. Modest fixes ⇒ >62 MOPS.



Full paper: usenix.org/conference/osdi26/presentation/schimmelpfennig

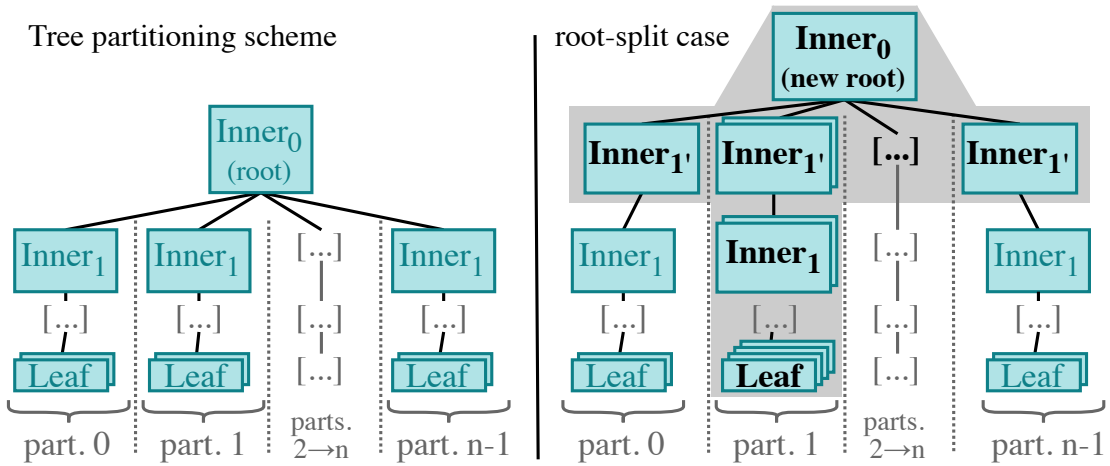
Thank you for your kind attention!

Stefan Lankes – slankes@eonerc.rwth-aachen.de

Institute for Automation of Complex Power Systems
E.ON Energy Research Center, RWTH Aachen University
Mathieustraße 10
52074 Aachen

www.acs.eonerc.rwth-aachen.de

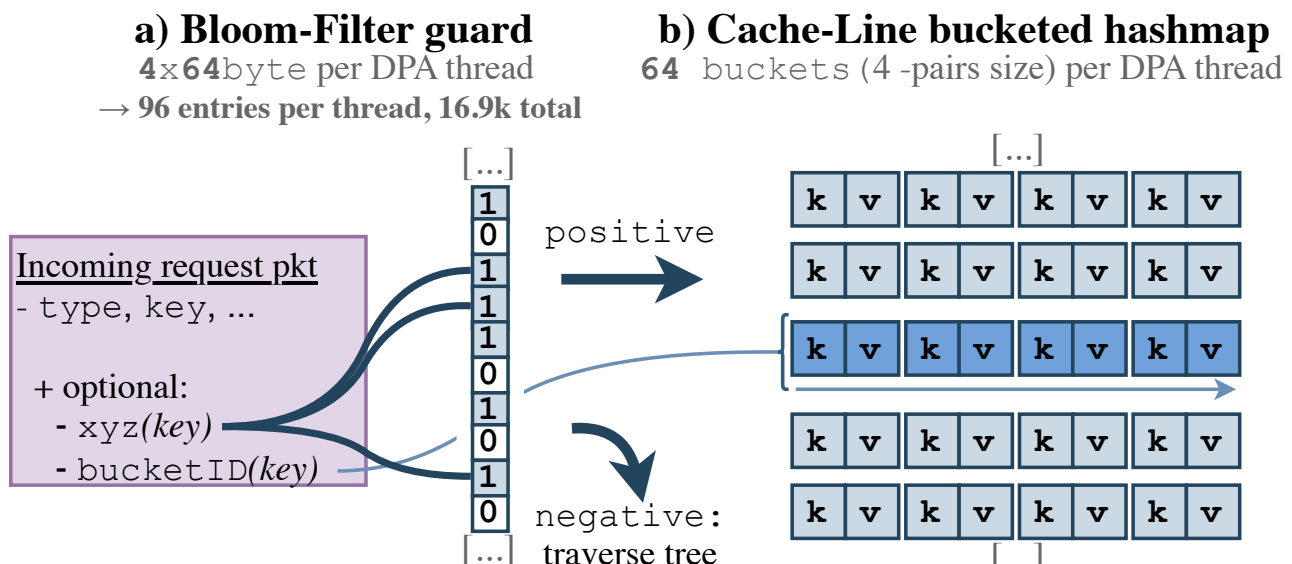
Backup: Stitching Updates into the NIC-Side Tree



- Multiple stitcher threads run on a dedicated DPA core.
- RCU semantics: COPY stitches prepare nodes; one atomic CONNECT stitch publishes the subtree.
- Epoch reclamation waits until every traverser has started a new request.

Backup: Hot-Entry Cache in Leftover Cache-Line Space

- Own hot cache per traverser thread.
  - Home thread per key: no cross-thread invalidation.
- 256-bit Bloom filter guard in spare thread-context cache-line space: check is free.
- ∼17 K entries cover >50% of requests (Zipf(1.0), 200 M keys). +**30% GET.**



Backup: What Makes ROLEX Stateful?

- ROLEX compute nodes are not just request senders; they participate in the index protocol.
- They keep a local learned-index cache, model metadata, leaf tables, synonym leaf tables, and remote-memory addresses.
- They execute modification protocol steps: one-sided RDMA reads/writes, CAS/FAA, leaf locking, synonym-leaf allocation, and leaf-table updates.
- They also handle stale-vs.-retrained model synchronization.
- DPA-Store clients only send UDP requests; index state and consistency work stay in the NIC/host system.

Backup: UDP, Loss, Retries, Idempotence

- UDP keeps the DPA-side protocol small; TCP per-flow state is far beyond the DPA cache/ context budget.
- GET and RANGE are naturally retryable.
- Write linearization point = insert-buffer append.
- Duplicate writes require idempotence at the application level; no exactly-once writes on the NIC path.

Backup: Overload and Flow Control

- Each traverser has a bounded queue (256 packets per thread).
- Clients use the response ratio as a simple back-pressure signal.
- Non-home rerouting bypasses the hot cache $\rightarrow$ fallback, not default.
- Full congestion control on the DPA remains an open problem.

Backup: Memory Overhead

- Index overhead 23–104% depending on dataset and ϵ.
- Larger ϵ (e.g., 16) reduces overhead on `face`/`osmc`.
- Comparison: B$^+$-tree $\approx$ 145%; ROLEX shifts index cost to client caches.
- Deliberate trade: spend NIC memory to remove client state.

Backup: B3220 vs. B3140L

- Uniform GET almost unchanged — memory-latency bound.
- Ping $+69\% \rightarrow$ stronger packet-matching throughput.
- Skewed GET: 48.5 MOPS with the hot cache.