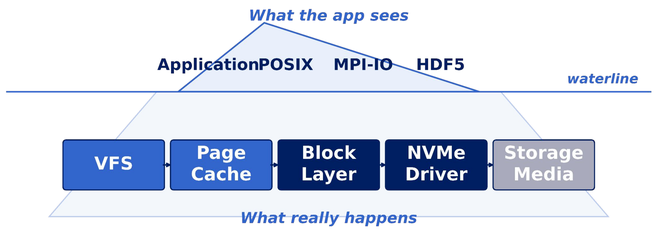
Extending I/O Observability to Explainability for Targeted Optimization

Neeraj Rajesh, Sarah Neuwirth, Jay Lofstead, Anthony Kougkas, Xian-He Sun, Nikoli Dryden, Hariharan Devarajan

HPC I/O stacks are opaque. Darshan counts POSIX operations but stops at the syscall boundary. Thousands of ranks descend through VFS, page cache, and block layer with no end-to-end trace. PEARL is an eBPF-based cross-layer tracing plugin for DataCrumbs. It builds one causal timeline per request, from syscall to disk completion. Layer-by-layer attribution shows where latency hides below the API enabling is targeted evidence based optimization.

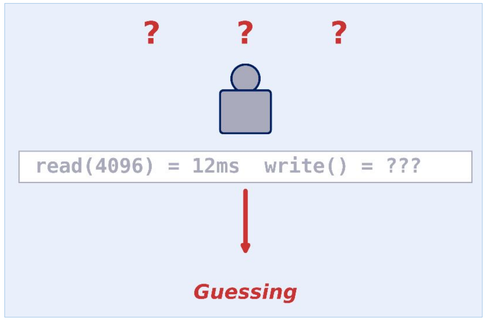
The I/O Explainability Iceberg: What is it & Why it matters?

- I/O is an Iceberg, limited visibility, most below kernel.
- Problem compounds in HPC scales.
- Every I/O request goes through VFS, cache, NVMe.

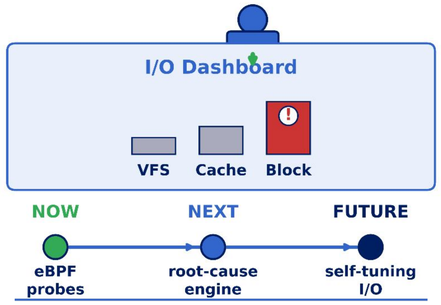


- Kernel tracing is not free, wasting time and money.
- Up to 41% overhead under high concurrency

- POSIX gives you latency, not its cause.
- Guessing is expensive: weeks spent chasing symptoms that live six layers below the API.



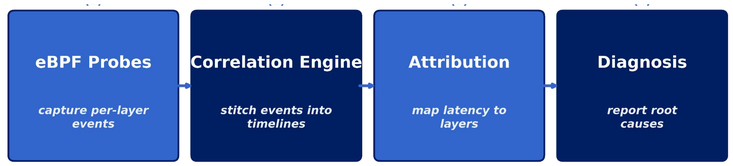
The Vision



In this poster we:

- **trace what POSIX cannot**: eBPF tracepoints for layers of the I/O stack.
- **causally attribute**: One causal timeline per request, attribution of IOPS

Achieving the Vision



- **eBPF probes at every layer**: VFS, cache, block.
- **Correlation engine stitches events**: Thread ID + timestamp + pointer matching fuses per-layer traces into one causal chain.
- **Attribution maps latency to layers**: Every millisecond assigned to its culprit: VFS, cache, or block.
- **Diagnosis with explainability:** Diagnosis report allows for deeper root cause analysis enabling evidence based optimization

The HPC Iceberg

The Gap: Comparing Darshan and PEARL

DARSHAN	PEARL
POSIX / MPI-IO counters	VFS → cache → block → NVMe tracepoints
Per-rank aggregates	Per-request causal timelines
Stops at syscall	End-to-end, syscall to Block I/O
What happened	Why, and where to fix

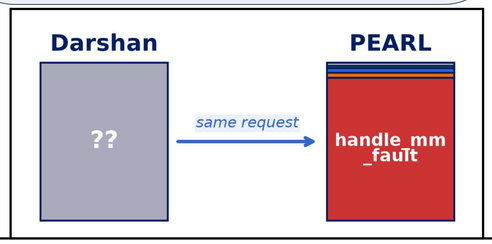
- **The result is opaque**: `read()` being 12 ms is recorded in Darshan, cannot say which layer caused stalled.
- **The gap compounds at scale**: Thousands of ranks, each with the same hidden descent. Counters are aggregated hiding the causation.

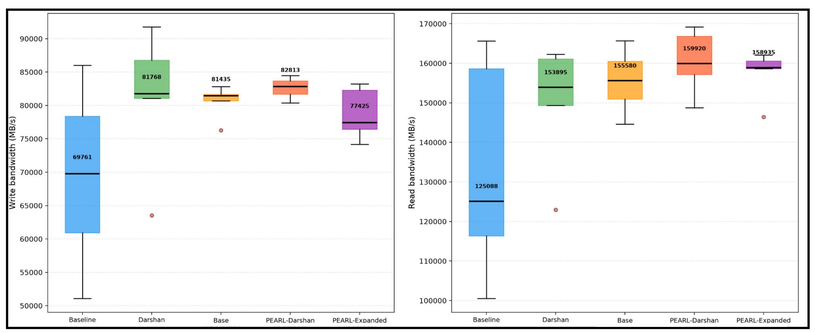
What the Data Shows

Experiment Goal: Difference in fidelity DARSHAN vs PEARL

- Chameleon Cloud, gigaio node
- OS: Ubuntu 26.04
- Storage: nvme
- Software:
  - MPI: OpenMPI
  - DARSHAN
  - PEARL
  - IOR

IOR Configuraion	Description
fsync-heavy	VFS → cache → block → NVMe tracepoints
large-direct	Per-request causal timelines
small-buf	End-to-end, syscall to Block I/O





- Our overheads are a negligible, variance is within system noise
- Page-cache faults account for 82% to 93% of kernel I/O time across tested workloads.
- In each case, **Darshan** reports POSIX write latency between 0.45 ms and 5.88 ms. **PEARL** traces the same requests through VFS, page cache, block layer, and NVMe, revealing total kernel time of 9.0 µs to 15.7 µs per request.
- The page-cache handle_mm_fault path consumes the vast majority of that time in every workload.

Lessons Learned

- Causal ordering is harder than collection. Matching bio to vfs_write requires thread+timestamp+pointer map.
- Tracing must trigger on outliers, not continuously. Full eBPF on 400K operations produces 530 MB traces.

Next Steps

- Detect anomaly, explain cause, optimize.
- Determine when to trigger detailed probe events.

Lawrence Livermore National Laboratory

LLNL-POST-2022625
Prepared by LLNL under Contract DE-AC52-07NA27344.

References:
Behzad, Babak, Surendra Byna, Prabhat, and Marc Snir. "Optimizing i/o performance of hpc applications with autotuning." *ACM Transactions on Parallel Computing (TOPC)* 5, no. 4 (2019): 1-27.

Snyder, Shane, Philip Carns, Kevin Harms, Robert Ross, Glenn K. Lockwood, and Nicholas J. Wright. "Modular hpc i/o characterization with darshan." In *2016 5th workshop on extreme-scale programming tools (ESPT)*, pp. 9-17. IEEE, 2016.

Devarajan, Hariharan, and Kathryn Mohror. "Extracting and characterizing I/O behavior of HPC workloads." In *2022 IEEE International Conference on Cluster Computing (CLUSTER)*, pp. 243-255. IEEE, 2022.

Bez, Jean Luca, Houjun Tang, Bing Xie, David Williams-Young, Rob Latham, Rob Ross, Sarp Oral, and Suren Byna. "I/o bottleneck detection and tuning: Connecting the dots using interactive log analysis." In *2021 IEEE/ACM Sixth International Parallel Data Systems Workshop (PDSW)*, pp. 15-22. IEEE, 2021.

Neuwirth, Sarah, Hariharan Devarajan, Chen Wang, and Jay Lofstead. "XIO: Toward eXplainable I/O for HPC Systems." In *Proceedings of the 37th International Conference on Scalable Scientific Data Management*, pp. 1-6. 2025.

National Nuclear Security Administration