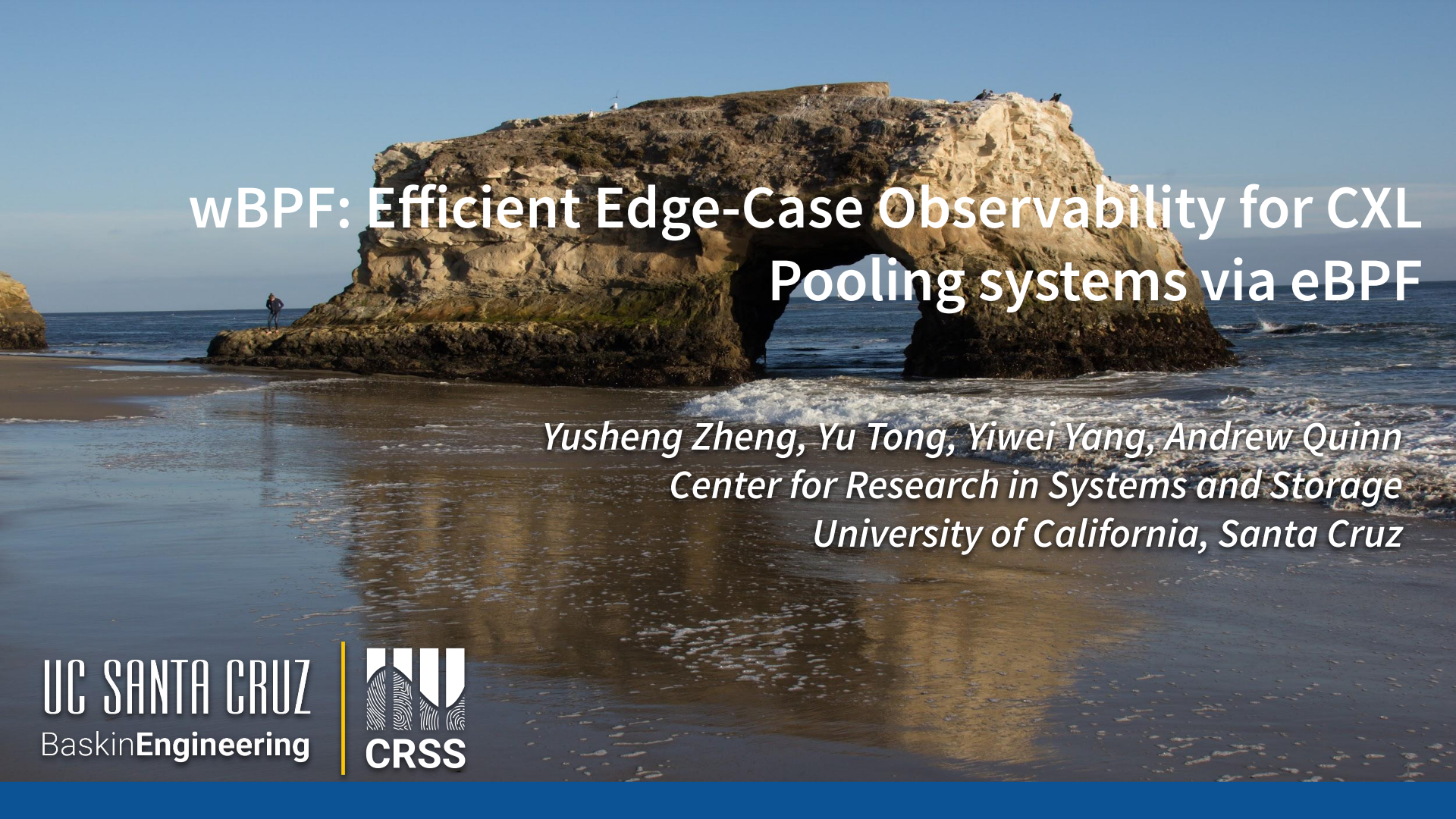


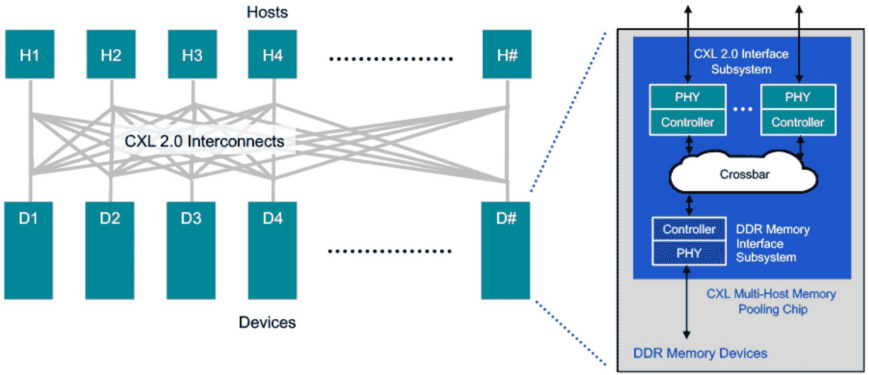
wBPF: Efficient Edge-Case Observability for CXL Pooling systems via eBPF

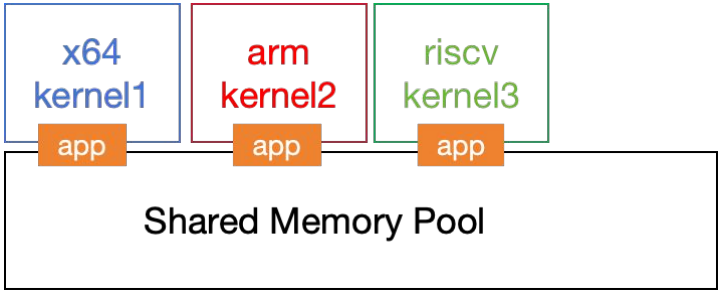
Yusheng Zheng, Yu Tong, Yiwei Yang, Andrew Quinn
Center for Research in Systems and Storage
University of California, Santa Cruz

UC SANTA CRUZ
BaskinEngineering
CRSS

CXL Pooling system is booming…

- High Level idea:
  - Use cache coherency to replace distributed socket calls.
  - No serialization and deserialization cost
  - CXL3.0 memory pooling enables multiple hosts to share pooled memory devices via low-latency interconnects.

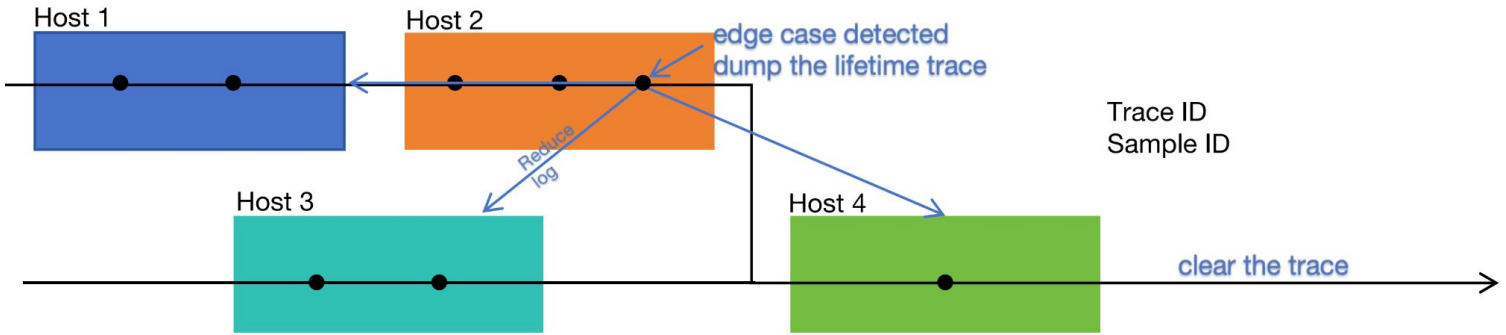




- Global I/O-free Shared Memory Objects (GISMO) by MemVerge
- Concord a FaaS distributed Caching Scheme over CXL by UIUC Jovan

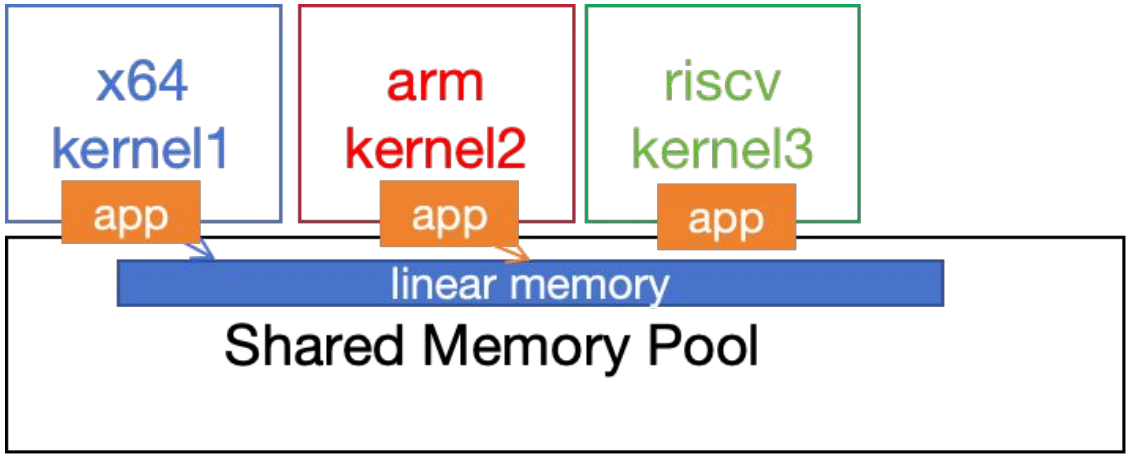
Challenges CXL memory pooling faces

- Unexpected tail latency from CXL pooling (cite).
- Concurrency bugs caused by multi-host memory environments, especially when using multiple memory models (e.g., ARM and x86) (cite)
- Existing academic (e.g., Hindsight) and industrial (e.g., Jaeger) tracing tools are designed for distributed systems and are incompatible with the shared memory setup of CXL pooling.



wBPF Design Overview

- Use lightweight language virtualization, in the form of WebAssembly, to provide a unified platform across heterogeneous systems.
  - Solves the concurrency and consistency problems that arise from sharing memory across heterogeneous hosts.
- Use a on-demand tracing library that captures detailed traces in a small ring buffer locally, and only outputs when anomalies occur.
- It is a fully automated system (i.e., it detects what to put in the ring buffer)



Semantic Probing

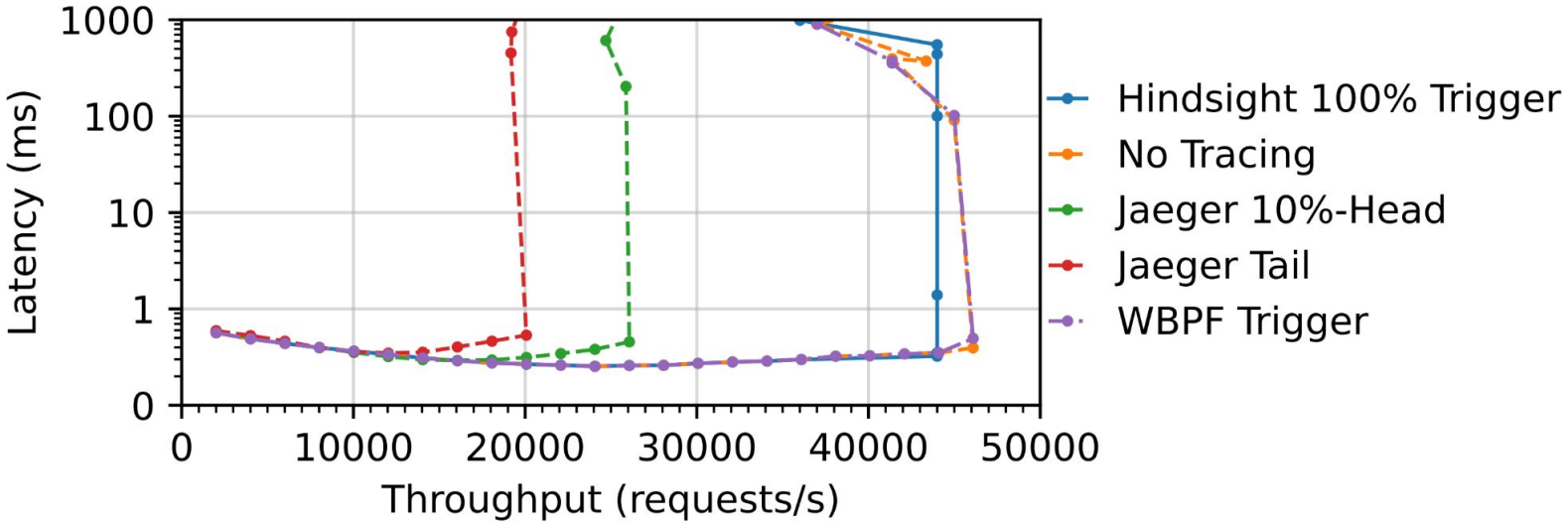
- 1. How does wBPF know what to put in the ring buffer? Applications use a variety of APIs that wBPF needs to support!
- 2. How does wBPF know when an anomaly occurs? The notion of an anomaly is application specific!

- Proposed solution: LLMs!
  - LLM suggests what parts of the program to trace and what relates to an anomaly
  - Run program in testing environment and collect traces.
  - pass traces to LLM to see if it is satisfied that the traces explain the anomalies.
  - Repeat as necessary.

Use Case: Low Overhead Tracing

- **Machine: single socket epyc 7742 with 256GB DDR4 Memory**
  Workloads: 2 services MicroBrick benchmarks

- **Minimal Performance Impact**
- **Throughput vs Latency**



Future Work

- **Scalability** – deploying wBPF in larger, multi-pool environments and ensuring its overhead stays predictably low as trace volume increases.
- Enriching the **semantic analysis**, possibly using more advanced AI models to cover a broader range of invariant violations.
- **Integrating** wBPF with existing distributed tracing ecosystems (like OpenTelemetry), so that memory pool traces can be correlated with application-level traces.
- As CXL technology evolves (e.g., CXL 3.0 with memory sharing between hosts), wBPF could be adapted to trace new usage models, making it a continuous asset for **future heterogeneous, composable systems**.