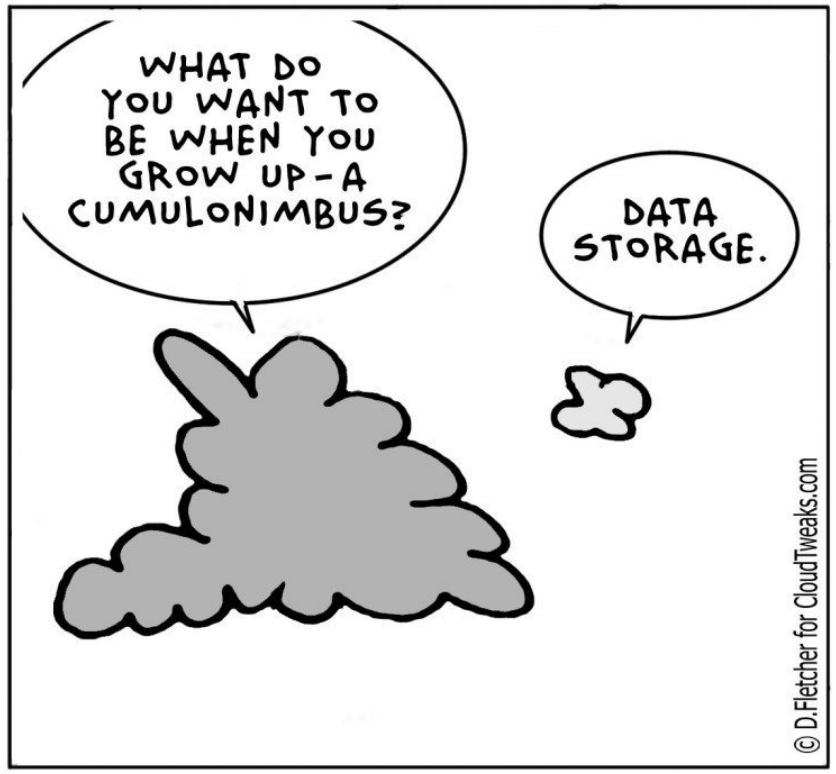
ConcurORAM: High-Throughput Stateless Parallel Multi-Client ORAM

Anrin Chakraborti
Stony Brook University

Radu Sion
Stony Brook University

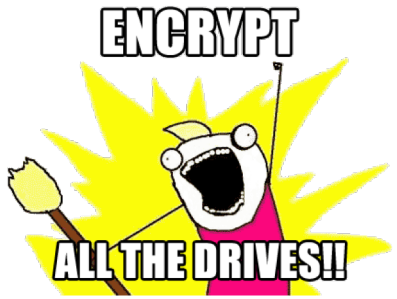
Its All About the Clouds!

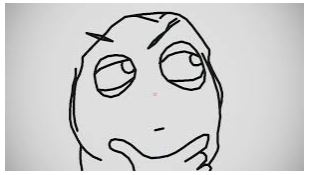


amazon web services™

Microsoft Azure

Protecting Outsourced Data



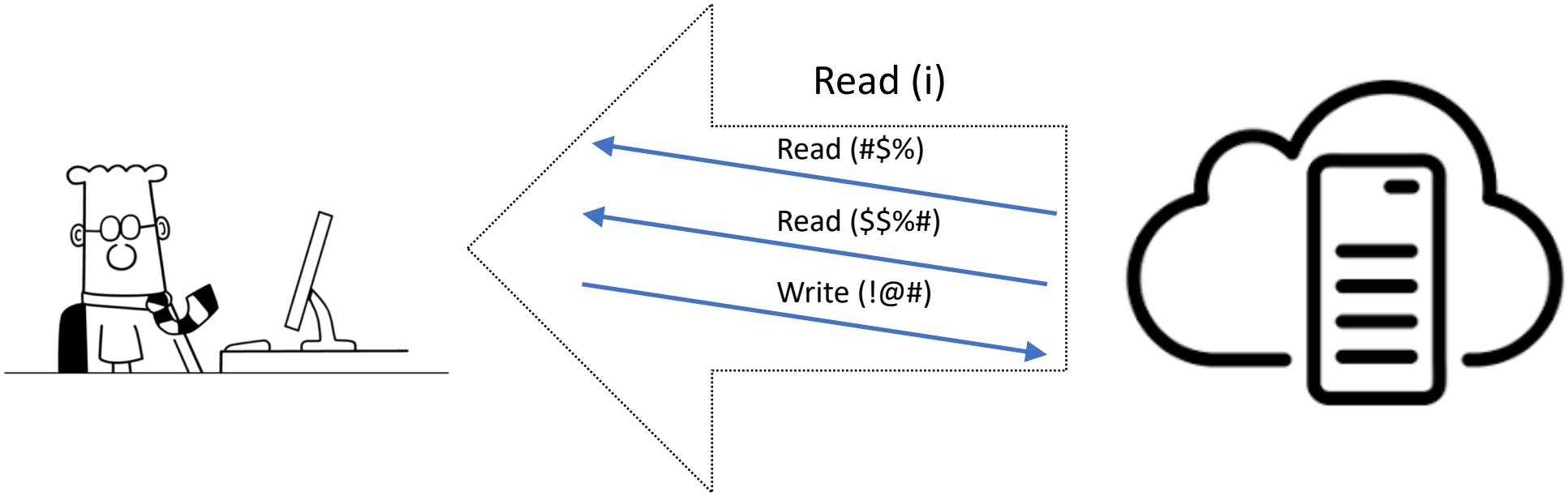


Read (i)

Write (k)

- Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. Islam et al. NDSS, ‘12
- Connecting the Dots: Privacy Leakage via Write-Access Patterns to the Main Memory. John et al. HOST, ’17
- …

Oblivious RAM (ORAM)



Observing the physical memory accesses, an adversary cannot learn

1. Which item has been accessed.
2. What operation has been performed.

Path ORAM [Stefanov et al. CCS ‘13]

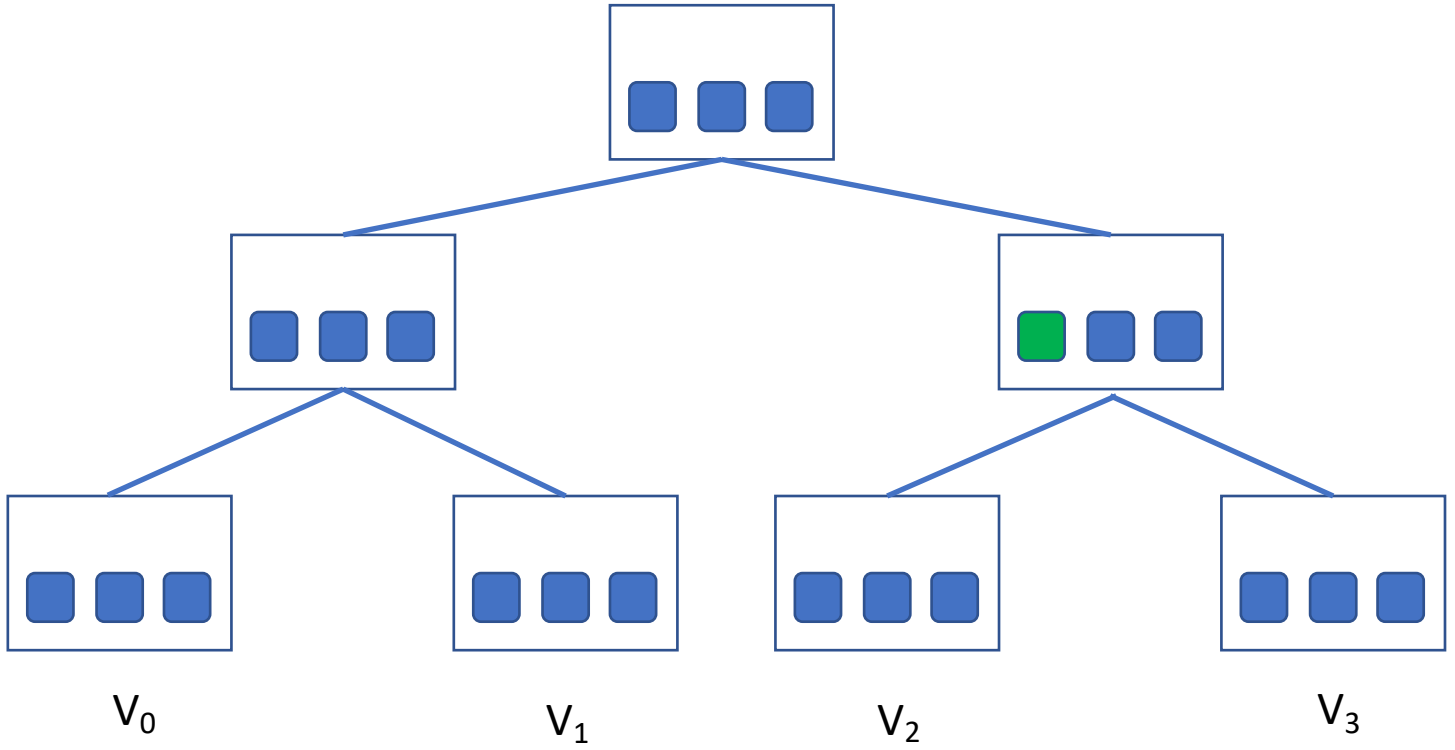
CLIENT

Position Map

BlockID	LeafID
0	v_2
1	v_0
2	v_3
…	…

STASH

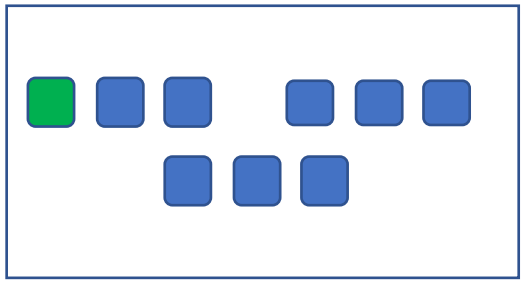
SERVER



Path ORAM Evictions

CLIENT

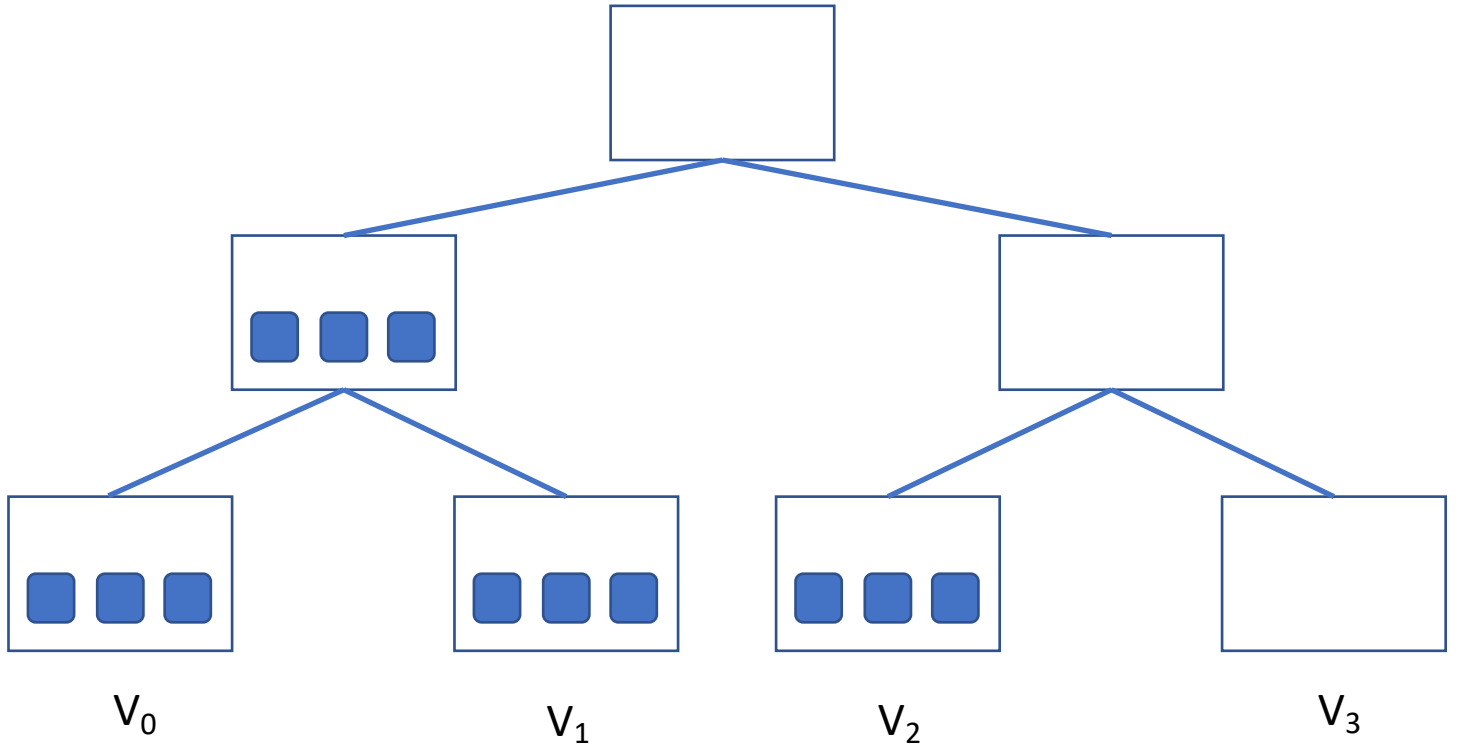
STASH



Position Map

BlockID	LeafID
0	v_2
1	v_0
2	v_3
…	…

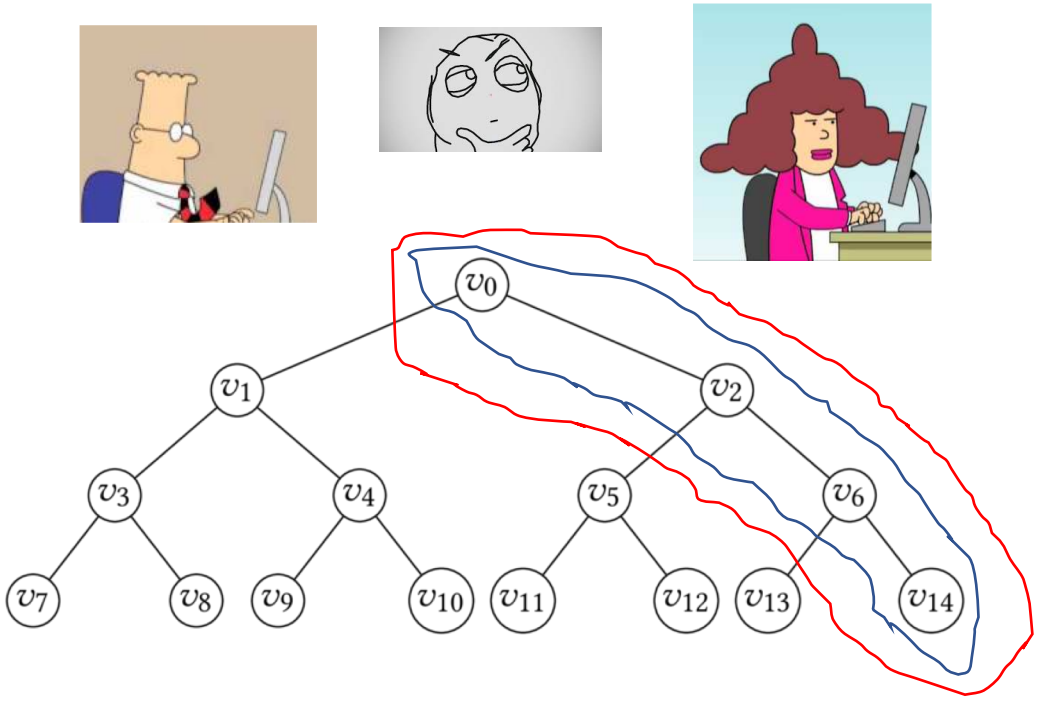
SERVER



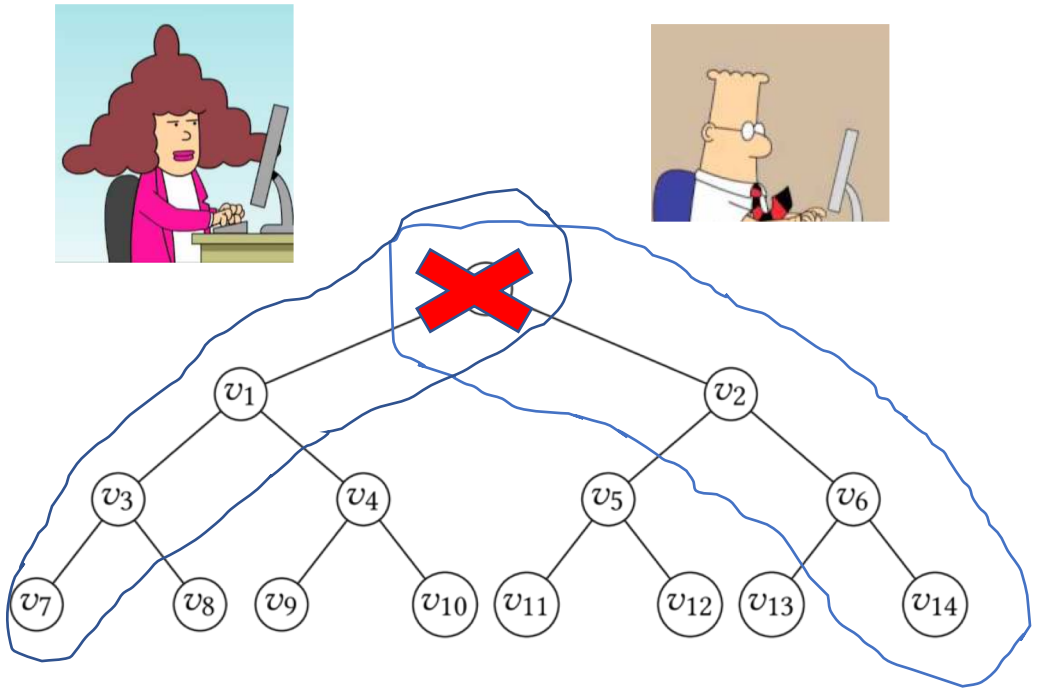
Can also evict along pre-determined paths

Multi-Client Scenarios

Inter-Client Privacy



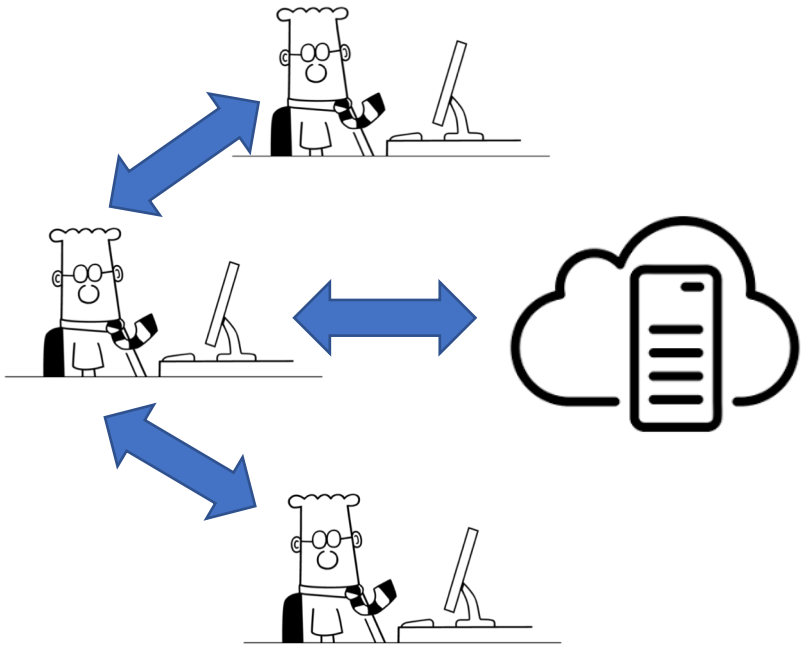
Correctness



Previous Approaches

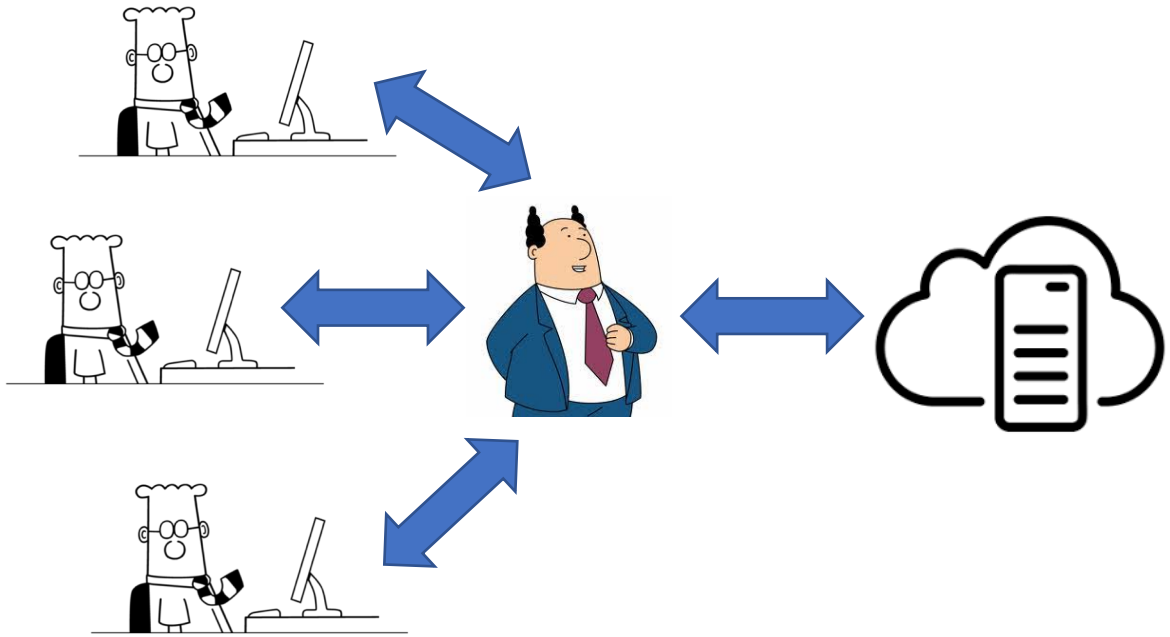
OPRAM: Inter-Client Comm.

- Rarely suitable for practical scenarios
- Clients need to be aware, online



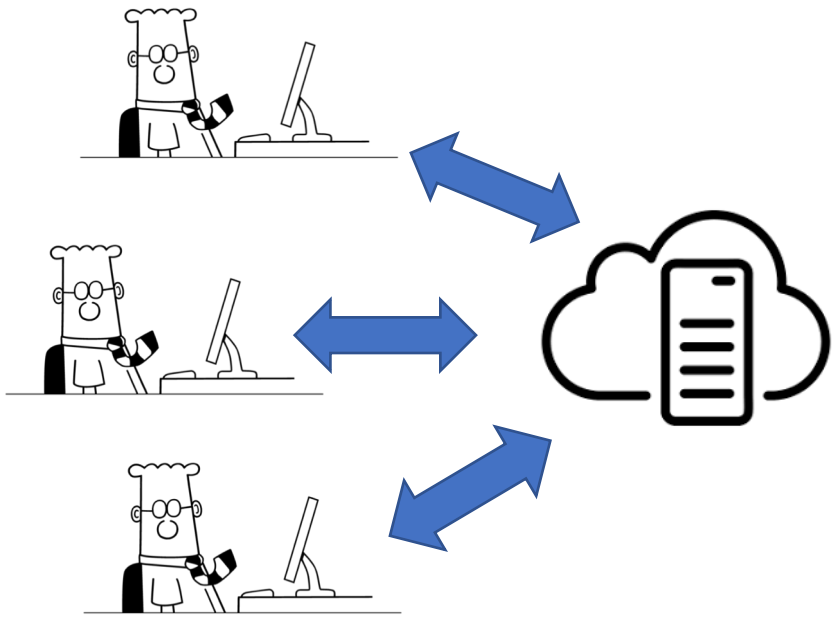
TaoStore: Trusted Proxy [S&P ’15]

- Violates ORAM trust model
- Proxy is a throughput bottleneck.
- Single point of failure/compromise



ConcurORAM Highlights

- ✓ No inter-client communication
- ✓ Eliminates proxy, scales better



Parallel Queries

Non-Blocking Evictions

Parallel Evictions

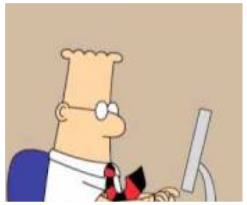
Parallel Queries

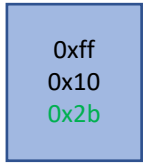
Query Log:

Transaction log with query addresses

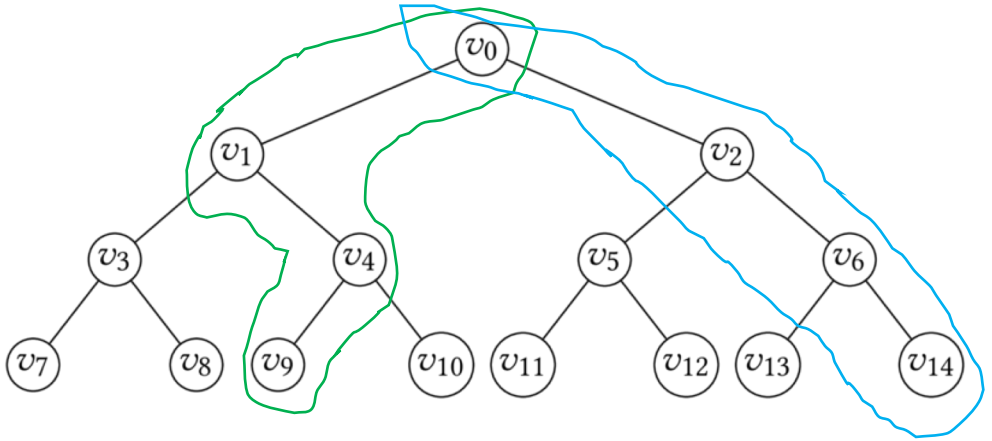
Result Log:

Obliviously cache query results



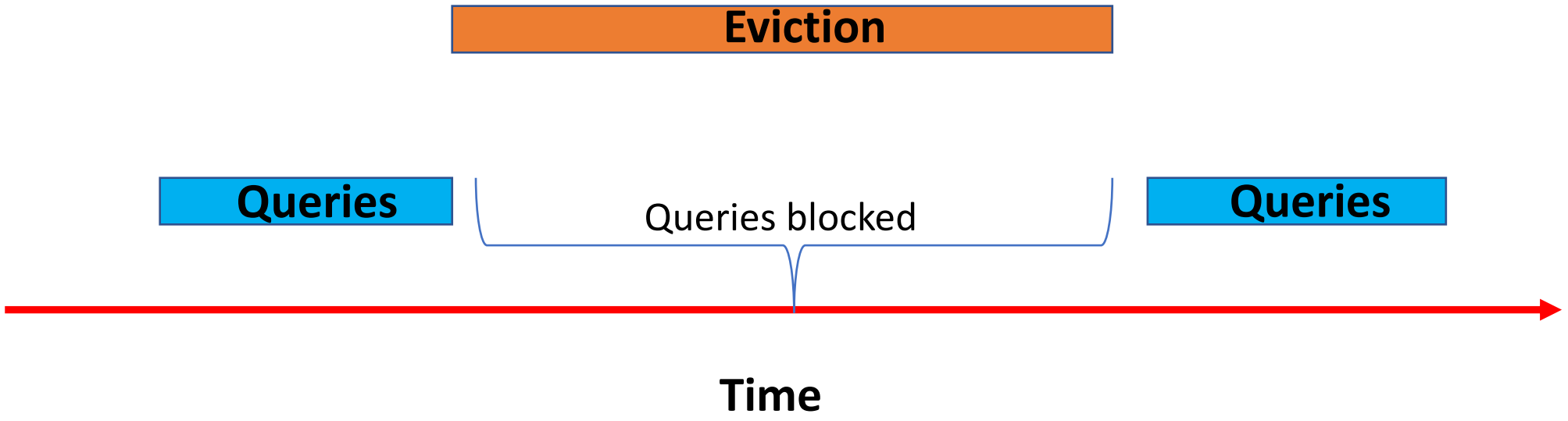


Query Log



Result Log

Path ORAM Timeline



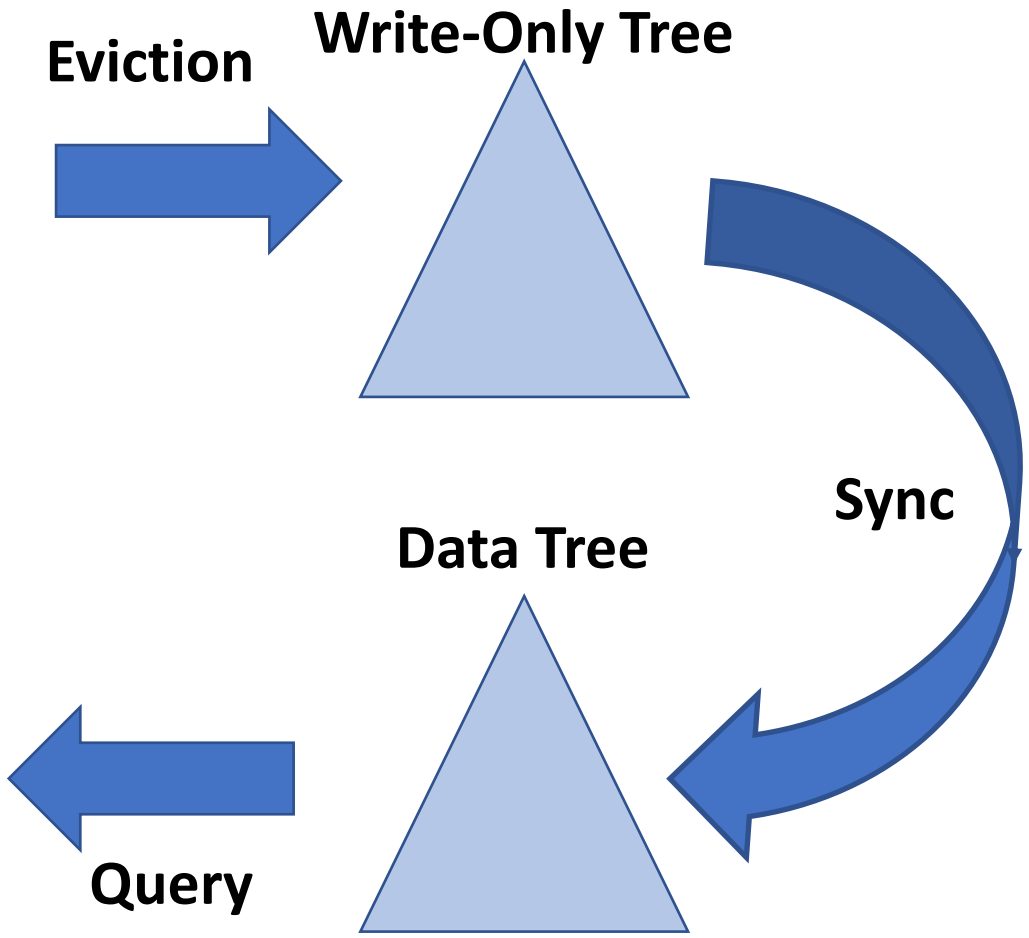
Non-Blocking Evictions

Problem:

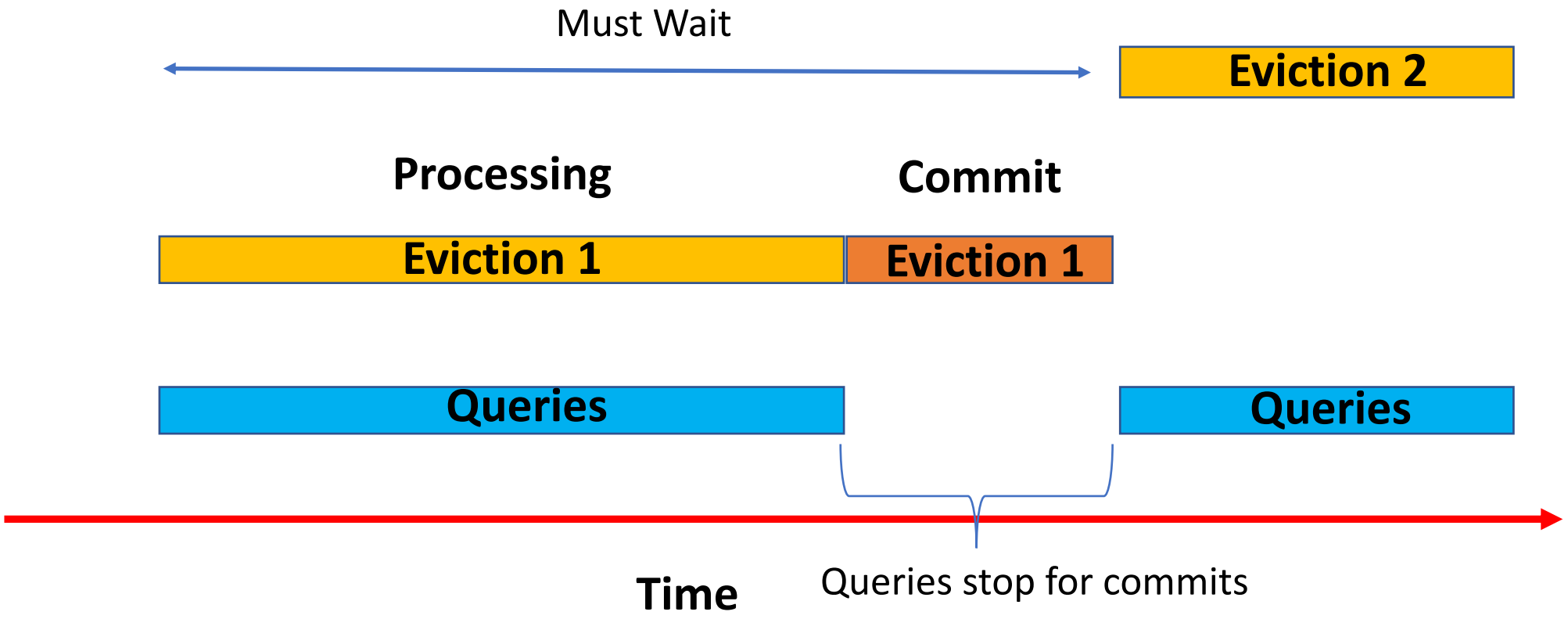
Evictions and queries access same data structures

Insight:

- Designated trees for queries and evictions
  - Sync periodically
- Multi-Phase Evictions:
  - **Processing** (Expensive): Update Write-only tree
  - **Commits**: Sync data tree



Non-Blocking Eviction Timeline



Parallel Evictions

Problem:

One evictions at a time is not!

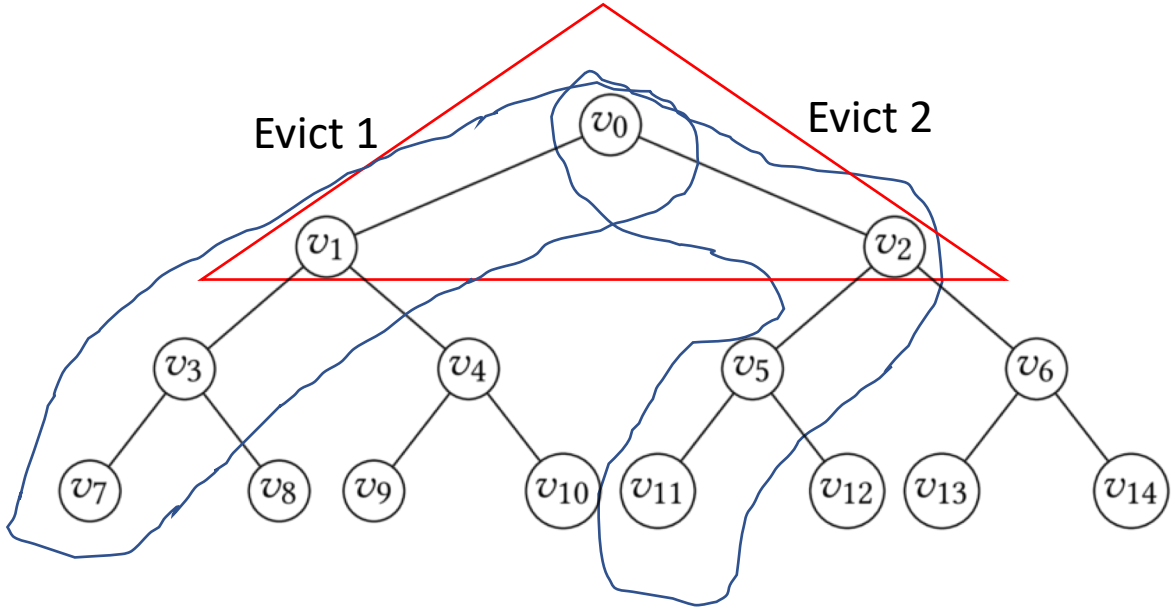
- Maintain consistency

Insight:

Deterministic eviction path selection

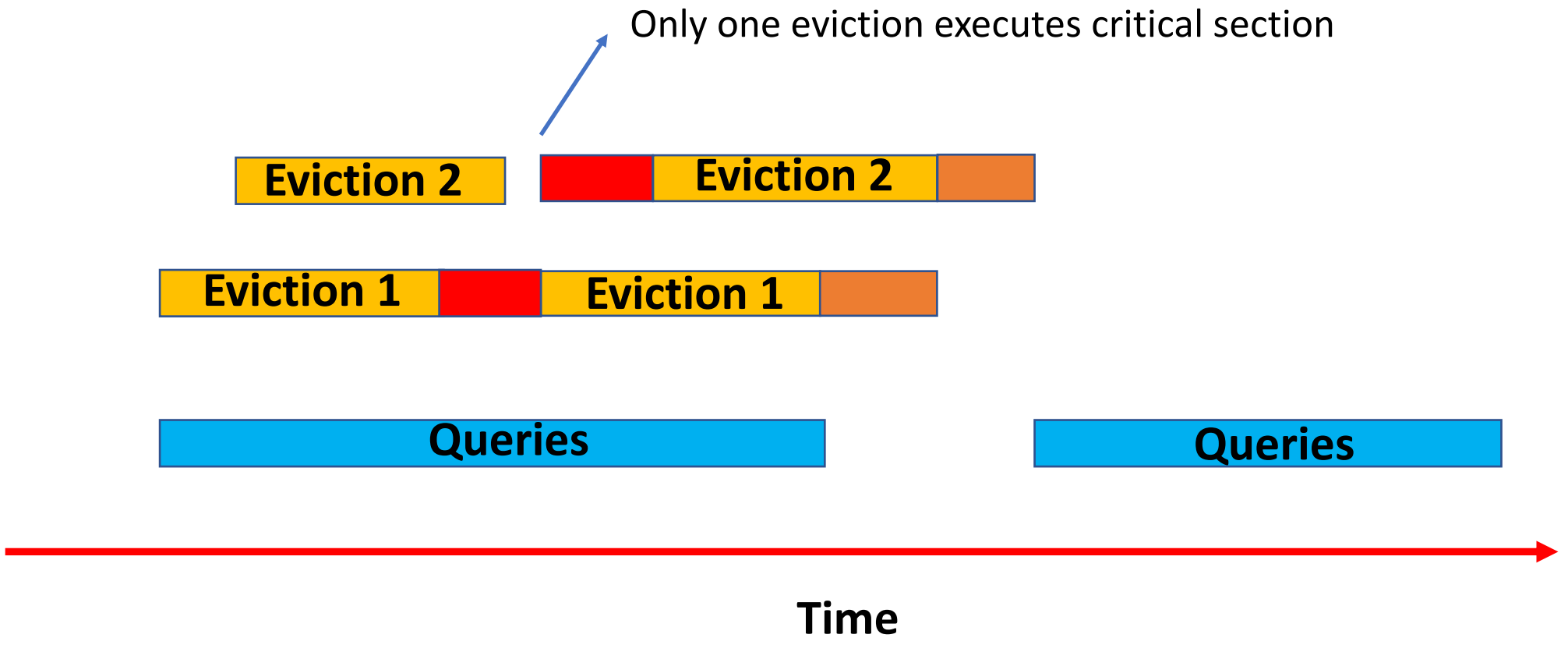
- fix number of evictions in parallel
- Fine-grained locks

Critical Section: Accessed serially and atomically



"If k consecutive eviction execute in parallel, the eviction paths intersect at only up to $\log(k) + 1$ levels of the tree"

Parallel Eviction Timeline

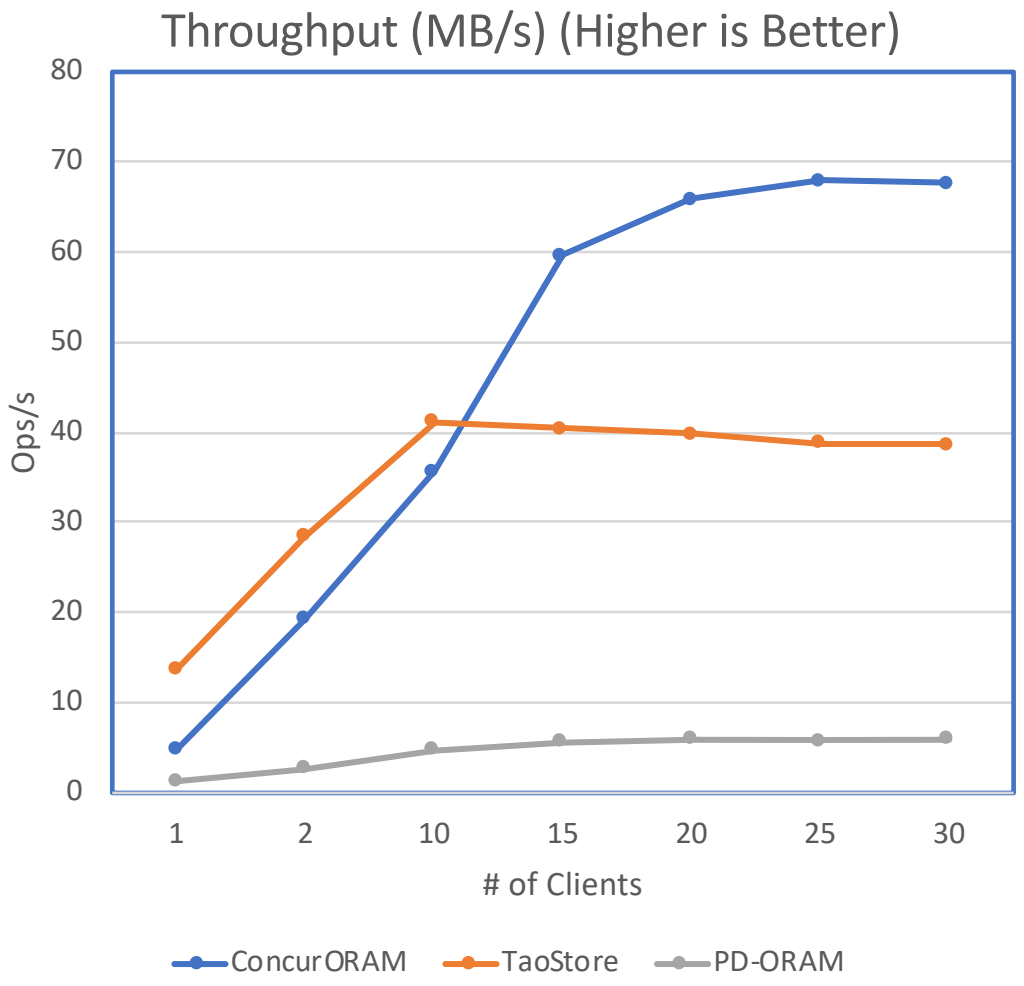


Other Challenges

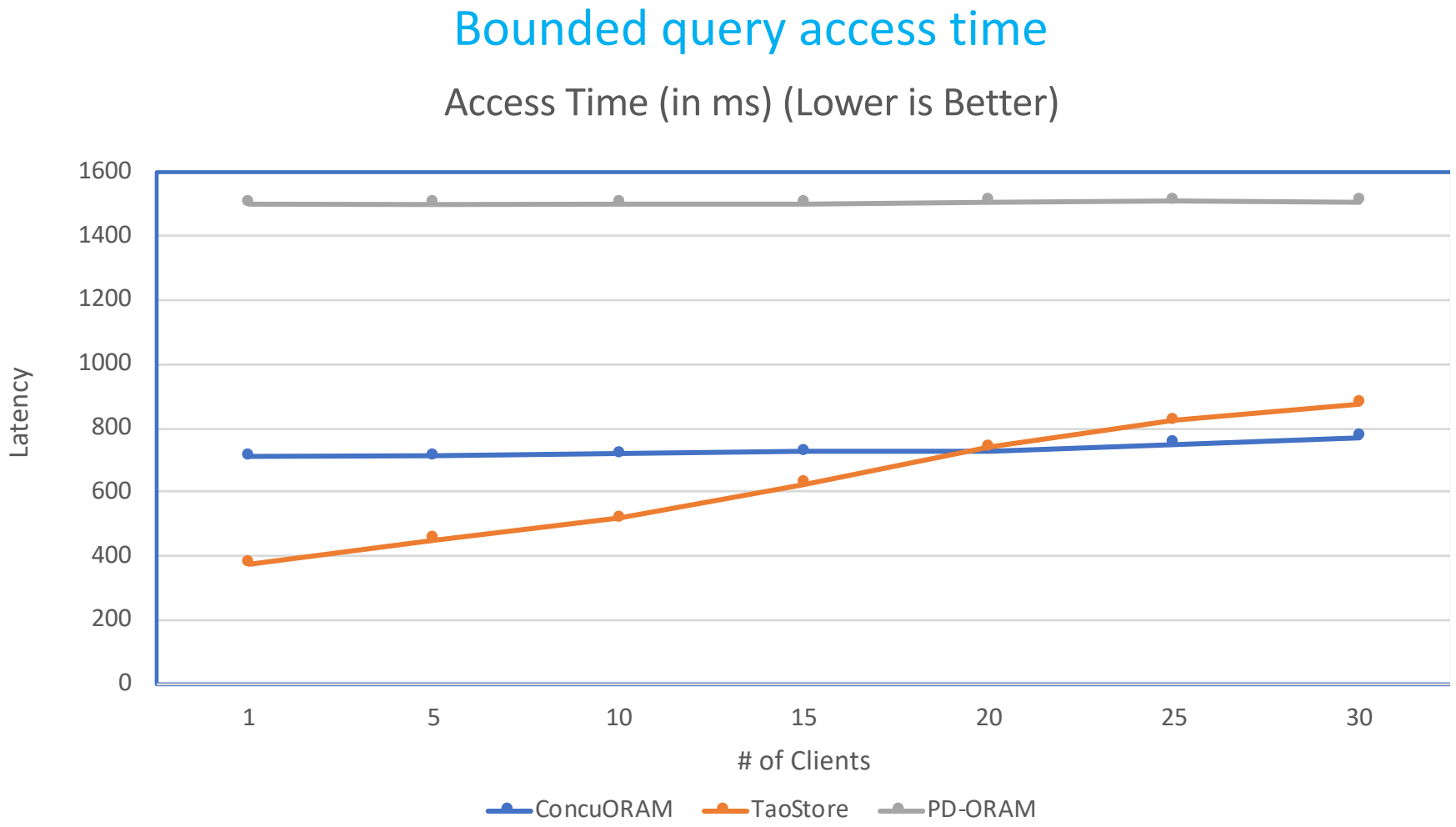
- Ordering eviction commits
- Store and privately query results
- Maintain metadata consistency
- Fault tolerance
- ….

Throughput

Throughput scales better



Query Latency



Summary

First Tree-Based Parallel ORAM

- No inter-client communication
- No Trusted Third-Parties

Parallelize Queries & Evictions

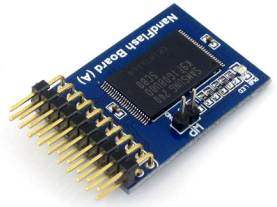
Scales Well

- 2x Overall Throughput

What I am working on

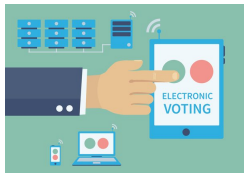
I am on the job market!

Oblivious RAM [NDSS ’19, ‘19]



Plausible Deniability [PETS ‘17, ‘19]

Integrity-Preserving Block Storage [ApSys ‘17]



History Independence [TIFS ’15]

Secure CPU Architecture & Secure Virtualization

Query Authentication [TKDE]

Thank you!!