Samurai: Slash your Decentralized Storage

Shistata Subedi*, **Asim Nepal***, Suyash Gupta*

*University of Oregon,

Why Historical Queries?

Decentralized systems allow users to collaboratively manage data, but **verifying the past** remains a challenge.

Real-world use cases:

- **Auditing**: Was a transaction included in block N?
- **History:** Was the balance more than $X between blocks M & N?
- **Legal Proofs:** Can you prove account status at a past block?

Existing systems hit the following bottlenecks:

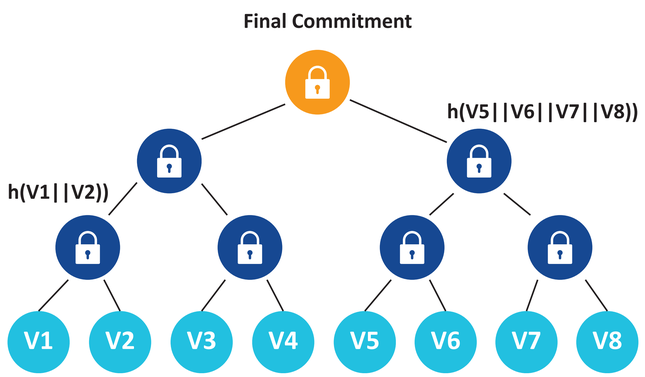
- **Storage Overhead**: Full history increases node size by ~25%.
- **Performance Hits:** Queries are slow and prone to DoS attacks.
- **Trust Leaks:** Relying on centralized explorers undermines decentralization.

Goal: Succinct, scalable, verifiable access to blockchain history.

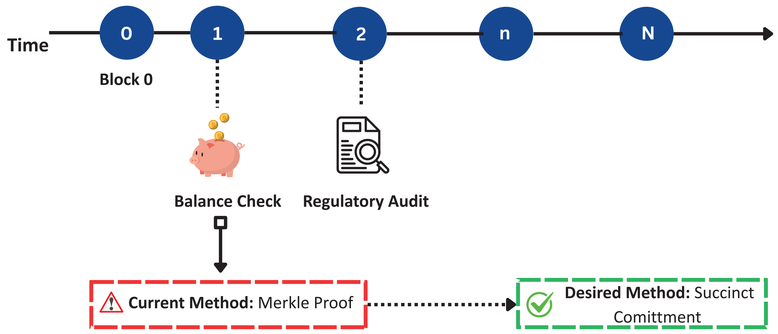
Our Approach: Samurai

- Recursive commitments over the account history.
- Support range queries over time.
- Ensure O(log t) proof generation costs.
- Witness size: small, even across millions of blocks

Samurai commits to structured aggregations of past values using a recursive construction.



Challenges



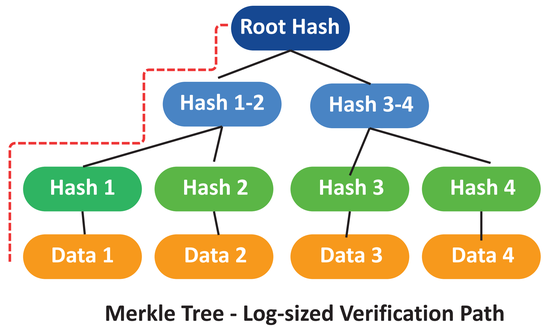
Permitting historical querying leads to the following tradeoffs:

Merkle trees: Well-known, log-sized proofs, but require full reconstruction for verification.

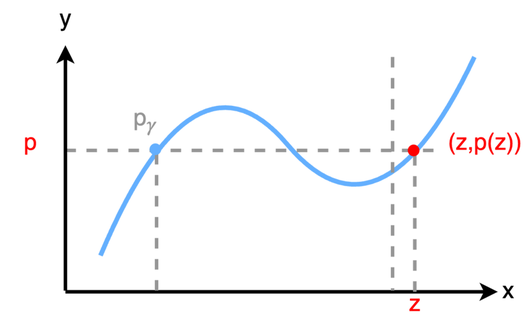
KZG commitments: Constant-size proofs, but integrating them over dynamic history is difficult.

Historical Queries Demand More:

- Efficient Proof Generation
- No Full State Replay
- Auditability with Minimal Overhead

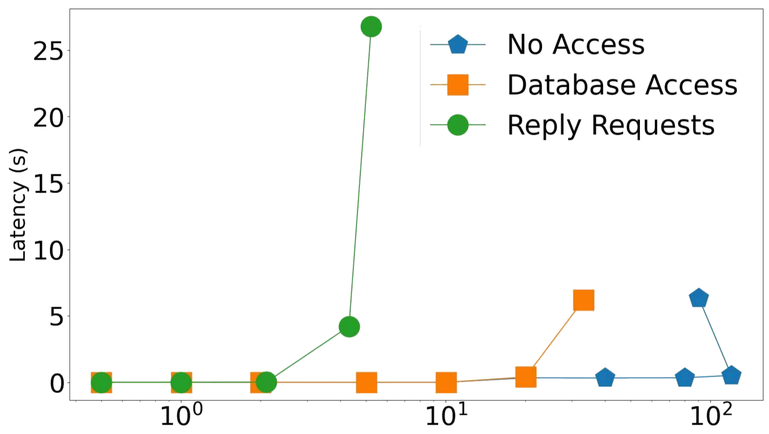


Merkle Tree - Log-sized Verification Path



KZG Commitment - Constant-sized Proof for any point

Preliminary Results



6x higher throughput vs. on-the-fly compute.

Open Questions

- Best structure for time evolving commitments?
- How to compress witnesses for unchanged values?
- Proof aggregation across long histories?

Conclusions

- Design a verifiable archival layer for decentralized systems using polynomial commitments.
- Enable efficient, auditable queries across long blockchain history.
- Reduce prover and verifier effort.
- Scale to millions of blocks (ever growing chain).