Permissioned Blockchain Through the Looking Glass: Architectural and Implementation Lessons Learned

Suyash Gupta
Sajjad Rahnama
Mohammad Sadoghi

MokaBlox LLC
Exploratory Systems Lab
University of California, Davis

Proceedings of the 40th IEEE ICDCS 2020
What is Blockchain?

- A linked list of blocks.
- Each block contains hash of the previous block.
- A block contains information about some client transactions.
Components of a Blockchain System

• Replicas → Store all the data.
• Client → Sends transactions to process.
• Consensus Protocol → Helps ordering transactions.
• Cryptographic Constructs → Authenticate replicas and clients.
• Ledger → Records transactions.
Famous Blockchain Applications?

Bitcoi

Iota

Algorand

PeerCoin

Dogecoin

Litecoin

Ripple

Ethereum
Why only Cryptocurrencies?

• Throughput of initial cryptocurrencies \( \rightarrow < 10 \text{ txns/s.} \)

• Throughput of existing distributed databases \( \rightarrow 1 \text{ million txns/s.} \)

• Low throughput acceptable in *permissionless* applications.

• **Aim:**  1) Cryptocurrency that is decentralized.

    2) Identities are hidden or unknown.

• **Result:**  1) Forks in the chain.

    2) Not acceptable to industries.
Rise of Permissioned Blockchains

- Only a selected group of replicas, although untrusted can participate.
- Identities of the replica known a priori.
- Prevent **chain forks**.
- Suitable for needs of an industry → JP Morgan, IBM, Oracle
- Open design of *Blockchain Databases*.
- Throughput? < 10K txns/s.
- Often cited reason → Traditional BFT consensus protocols are expensive!
At the core of any Blockchain application is a Byzantine Fault-Tolerant (BFT) consensus protocol.
Can a well-crafted system based on a classical BFT protocol outperform a modern protocol?

ResilientDB employs three-phase (of which two require quadratic communication) PBFT protocol and scales better than protocol-centric permissioned blockchain system that uses single linear-phase Zyzzyva.
Existing Permissioned Blockchain systems overlook system design!

ResilientDB adopts *well-researched* database and system practices.

Visit at: [https://resilientdb.com/](https://resilientdb.com/)
Dissecting existing Permissioned Blockchain

1) Single-threaded Monolithic Design
2) Successive Phases of Consensus
3) Integrated Ordering and Execution
4) Strict Ordering
5) Off-Chain Memory Management
6) Expensive Cryptographic Practices
ResilientDB Architecture

- **Secure Layer**: Hashing Toolkit, Signing Toolkit
- **Execution Layer**: Threads, BFT Consensus, Queues
- **Storage Layer**: Blockchain, Metadata
- **Network**
Multi-Threaded Deep Pipeline at Replicas

Flow:
1. **Input** (Message from Clients and Replicas)
2. **Client Requests**
3. **Prepare & Commit**
4. **Batch Creation**
5. **Worker**
6. **Checkpoint**
7. **Execute**
8. **Output** (Message to Replicas and Clients)

Network connections ensure the flow of messages through the pipeline.
Evaluation and Analysis

• We ask eleven distinct questions that affect performance of a Permissioned Blockchain.

• Workload provided by Yahoo Cloud Serving Benchmark (YCSB).

• PBFT to achieve BFT consensus among replicas.

• General Setup (unless stated otherwise):
  • 8-core Intel Xeon Cascade Lake CPU.
  • Requests sent by 80K clients deployed on 4 machines.
  • Employed batching \rightarrow Batch size set at 100.
  • At each replica \rightarrow one worker-thread, one execute-thread and two batch-threads
Insight 1: Multi-Threaded pipeline Gains

Parallelizing and Pipelining tasks across worker, execution (E) and batch-threads (B).
Insight 1: Multi-Threaded pipeline Gains

Parallelizing and Pipelining tasks across worker, execution (E) and batch-threads (B).
Insight 1: Multi-Threaded pipeline Gains

Parallelizing and Pipelining tasks across worker, execution (E) and batch-threads (B).
Insight 1: Multi-Threaded pipeline Gains

Parallelizing and Pipelining tasks across worker, execution (E) and batch-threads (B).
Insight 2: Optimal Batching Gains

More transactions batched together → increase in throughput
→ reduced phases of consensus.
Insight 3: Memory Storage Gains

In-memory blockchain storage → reduces access cost.
Insight 4: Number of Clients

Too many clients → increases average latency.
Conclusions and Final Remarks

• There are several factors that affect throughput of a blockchain system.

• Fast consensus does not always imply an efficient blockchain system.

• We show that a well-crafted system-centric permissioned blockchain system can outperform a protocol-centric blockchain system.

• System designers need to dissect their application to find performance bottlenecks.
Thank You