OmniLedger: A Secure, Scale-Out, Decentralized Ledger via Sharding

Lefteris Kokoris-Kogias (@LefKok)

Decentralized and Distributed Systems Lab (DEDIS)
EPFL, IBM PhD Fellow, IC3

Scaling Bitcoin
2018-10-07, Tokyo
Acknowledgements

Philipp Jovanovic  
(EPFL, CH)

Linus Gasser  
(EPFL, CH)

Nicolas Gailly  
(EPFL, CH)

Ewa Syta  
(Trinity College, USA)

Bryan Ford  
(EPFL, CH)
Talk Outline

• Motivation
• OmniLedger
• Evaluation
• Conclusion
Talk Outline

- Motivation
- OmniLedger
- Evaluation
- Conclusion
Scaling Blockchains is More Important Than Ever …

The ethereum network is getting jammed up because people are rushing to buy cartoon cats on its blockchain.
# Bitcoin vs OmniLedger

<table>
<thead>
<tr>
<th></th>
<th>Bitcoin</th>
<th>OmniLedger*</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Throughput</strong></td>
<td>~4 TPS</td>
<td>~20,000 TPS</td>
</tr>
<tr>
<td><strong>1-st Confirmation</strong></td>
<td>~10 minutes</td>
<td>~1 second</td>
</tr>
<tr>
<td><strong>Full Security</strong></td>
<td>~60 minutes</td>
<td>~42 second</td>
</tr>
<tr>
<td><strong>More Available</strong></td>
<td>No performance</td>
<td>Linear Increase in</td>
</tr>
<tr>
<td><strong>Resources</strong></td>
<td>Gain</td>
<td>Throughput</td>
</tr>
</tbody>
</table>

* Configuration with 1120 validators against a 12.5% adversary
# Bitcoin vs OmniLedger

<table>
<thead>
<tr>
<th>Feature</th>
<th>Bitcoin</th>
<th>OmniLedger*</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Throughput</strong></td>
<td>~4 TPS</td>
<td>~20.000 TPS</td>
</tr>
<tr>
<td><strong>1-st Confirmation</strong></td>
<td>~10 minutes</td>
<td>~1 second</td>
</tr>
<tr>
<td><strong>Full Security</strong></td>
<td>~60 minutes</td>
<td>~42 second</td>
</tr>
<tr>
<td><strong>More Available Resources</strong></td>
<td>No performance gain</td>
<td>Linear Increase in Throughput</td>
</tr>
</tbody>
</table>

* Configuration with 1120 validators against a 12.5% adversary

**Scale-Out**
What we Want: Scale-Out Performance

Scale-out: Throughput increases *linearly* with the available resources.
… But Scaling Blockchains is Not Easy
Distributed Ledger Landscape

L. Luu et al., A Secure Sharding Protocol for Open Blockchains, CCS 2016


G. Danezis and S. Meiklejohn, Centrally Banked Cryptocurrencies, NDSS 2016
No Scale-Out (Bitcoin)
Scale-Out (OmniLedger)

- How do validators choose which blockchain to work on?
- How can I pay a yellow vendor with greencoins?

Double Throughput
Random Validator Assignment

- Let validators choose? —> All malicious validators can choose the same chain

- Randomly assign validators? —> Preserve security for adequately large shard size
Bootstraping Identities

- **Key blocks:**
  - PoW & share value
  - Leader election

- **Microblocks:**
  - Validating client transactions
  - Issued by the leader
Strawman: SimpleLedger

Overview

• Evolves in epochs $e$
• Trusted randomness beacon emits random value $rnd_e$
• Validators:
  ‣ Use $rnd_e$ to compute shard assignment (ensures shard security)
  ‣ Process tx using consensus within one shard (ByzCoin)
Strawman: SimpleLedger

**Security Drawbacks**
- Randomness beacon: trusted third party
- No tx processing during validator re-assignment
- No cross-shard tx support

**Performance Drawbacks**
- ByzCoin failure mode
- High storage and bootstrapping cost
- Throughput vs. latency trade-off
Talk Outline

• Motivation

• OmniLedger

• Evaluation

• Conclusion
Roadmap

**SimpleLedger**
- Sharding via distributed randomness
- Smooth epoch transitions
- Atomix: Atomic cross-shard txs
- **Security**
  - ByzCoinX: Robust BFT consensus

**OmniLedger**
- Shard ledger pruning
- **Performance**
  - Trust-but-verify validation: Throughput / Latency trade-off
Roadmap

SimpleLedger

- Sharding via distributed randomness
- Smooth epoch transitions
- Atomix: Atomic cross-shard txs

Security

- ByzCoinX: Robust BFT consensus

Performance

- Shard ledger pruning
- Trust-but-verify validation: Throughput / Latency trade-off

OmniLedger
Roadmap

SimpleLedger

- Sharding via distributed randomness
- Smooth epoch transitions
- Atomix: Atomic cross-shard txs
- ByzCoinX: Robust BFT consensus

OmniLedger

- Shard ledger pruning
- Trust-but-verify validation: Throughput / Latency trade-off

Security

Performance
Shard Validator Assignment

1. Temp. leader election (Can be biased)

2. Randomness generation (Output is unbiasable)

3. Shard assignment (using $rnd_e$)

* Syta, Ewa, et al. "Scalable bias-resistant distributed randomness." Oakland ‘17
Roadmap

SimpleLedger

- Sharding via distributed randomness
- Smooth epoch transitions
- Atomix: Atomic cross-shard txs
- ByzCoinX: Robust BFT consensus

Security

Performance

OmniLedger

- Shard ledger pruning
- Trust-but-verify validation: Throughput / Latency trade-off
Two-Phase Commit

Coordinator

Server

Query to commit

prepare / abort

Commit / Rollback

commit / abort
**Atomix: Cross-Shard Transactions**

**Challenge:**
- Cross-shard tx commit atomically or abort eventually

**Solution:** Atomix
- Client-managed protocol
  1. Client sends cross-shard tx to input shards
  2. Collect ACK/ERR proofs from input shards
  
(a) If all input shards accept, commit to output shard, otherwise
(b) abort and reclaim input funds

The Atomix protocol for secure cross-shard transactions
Roadmap

SimpleLedger

- Sharding via distributed randomness
- Smooth epoch transitions
- Atomix: Atomic cross-shard txs

Security

- ByzCoinX: Robust BFT consensus

Performance

- Shard ledger pruning
- Trust-but-verify validation: Throughput / Latency trade-off

OmniLedger
Challenge:
• Latency vs. throughput trade-off

Solution:
• Two-level “trust-but-verify” validation
  • Low latency:
    ‣ Optimistically validate transactions by “insecure” shards
  • High throughput:
    ‣ Batch optimistically validated blocks and audit by “secure” shards
Talk Outline

• Motivation
• OmniLedger
• Evaluation
• Conclusion
Implementation & Experimental Setup

Implementation

• OmniLedger and its subprotocols (ByzCoinX, Atomix, etc.) implemented in Go

• Based on DEDIS code
  ‣ Kyber crypto library
  ‣ Onet network library
  ‣ Cothority framework

• https://github.com/dedis

DeterLab Setup

• 48 physical machines up to 1800 clients
  ‣ Intel Xeon E5-2420 v2 (6 cores @ 2.2 GHz)
  ‣ 24 GB RAM
  ‣ 10 Gbps network link

• Network restrictions (per client)
  ‣ 20 Mbps bandwidth
  ‣ 200 ms round-trip latency
Evaluation: **Scale-Out**

<table>
<thead>
<tr>
<th>#validators (#shards)</th>
<th>70 (1)</th>
<th>140 (2)</th>
<th>280 (4)</th>
<th>560 (8)</th>
<th>1120 (16)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OmniLedger (tx/sec)</td>
<td>439</td>
<td>869</td>
<td>1674</td>
<td>3240</td>
<td>5850</td>
</tr>
<tr>
<td>Bitcoin (tx/sec)</td>
<td>~4</td>
<td>~4</td>
<td>~4</td>
<td>~4</td>
<td>~4</td>
</tr>
</tbody>
</table>

**Scale-out** throughput for 12.5%-adversary and **shard size 70** and 1200 validators
Evaluation: Throughput

Results for 1800 validators
Evaluation: **Latency**

Transaction confirmation latency in seconds for regular and multi-level validation

<table>
<thead>
<tr>
<th>#shards, adversary</th>
<th>4, 1%</th>
<th>25, 5%</th>
<th>70, 12.5%</th>
<th>600, 25%</th>
</tr>
</thead>
<tbody>
<tr>
<td>regular validation</td>
<td>1.38</td>
<td>5.99</td>
<td>8.04</td>
<td>14.52</td>
</tr>
<tr>
<td>1st lvl. validation</td>
<td>1.38</td>
<td>1.38</td>
<td>1.38</td>
<td>4.48</td>
</tr>
<tr>
<td>2nd lvl. validation</td>
<td>1.38</td>
<td>55.89</td>
<td>41.89</td>
<td>62.96</td>
</tr>
<tr>
<td>Bitcoin</td>
<td>600</td>
<td>600</td>
<td>600</td>
<td>600</td>
</tr>
</tbody>
</table>

Latency increase since optimistically validated blocks are batched into larger blocks for final validation to get better throughput.
Talk Outline

• Motivation
• OmniLedger
• Experimental Results
• **Conclusion**
Conclusion

- **OmniLedger – Secure scale-out distributed ledger framework**
  - Atomix: Client-managed cross-shard tx
  - ByzCoinX: Robust intra-shard BFT consensus
  - Sharding: Visa-level throughput and beyond
  - Trust-but-verify validation: No latency vs. throughput tradeoff
  - For PoW, PoS, permissioned, etc.

- **Code:** [https://github.com/dedis](https://github.com/dedis)

- **Contact:** eleftherios.kokoriskogias@epfl.ch, @LefKok