Microsoft

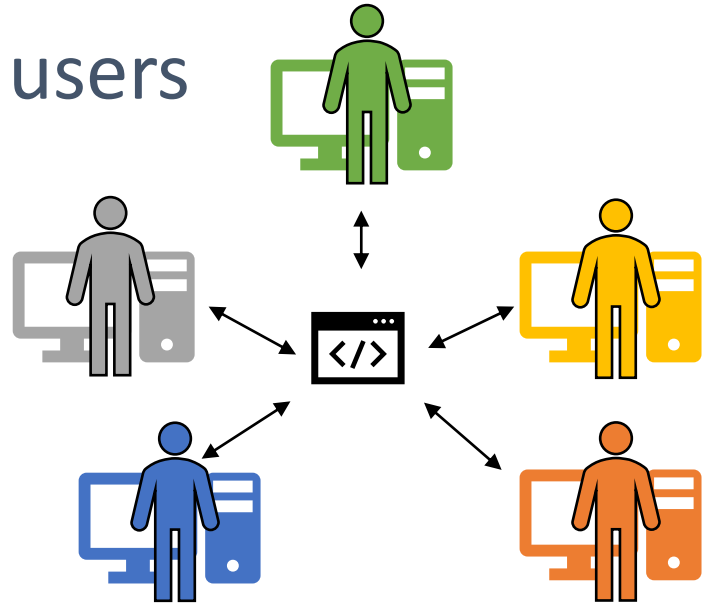
IA-CCF: Individual Accountability for Permissioned Ledgers

Alex Shamis[1,2], Peter Pietzuch[1,2], Burcu Canakci[3], Miguel Castro[1], Cedric Fournet[1], Edward Ashton[1], Amaury Chamayou[1], Sylvan Clebsch[1], Antoine Delignat-Lavaud[1], Matt Kerner[4], Julien Maffre[1], Olga Vrousgou[1], Christoph M. Wintersteiger[1], Manuel Costa[1], and Mark Russinovich[4]

[1]**Microsoft Research,** [2]**Imperial College London,** [3]**Cornell University,** [4]**Microsoft Azure**

What is a permissioned ledger?

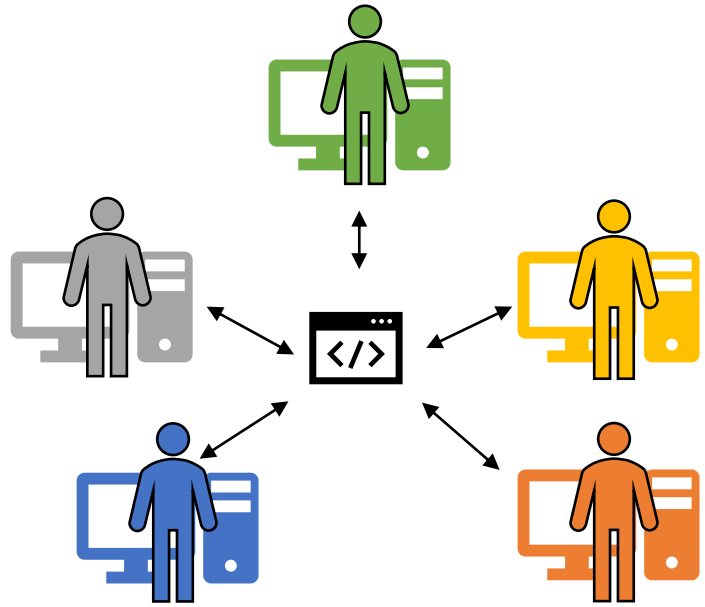
- Distributed system that provides a service for users
  - Record events in a ledger
  - Consortium members bring replicas
  - Replicas operated by consortium members



- *Only authorized users, members, and replicas can access the system*

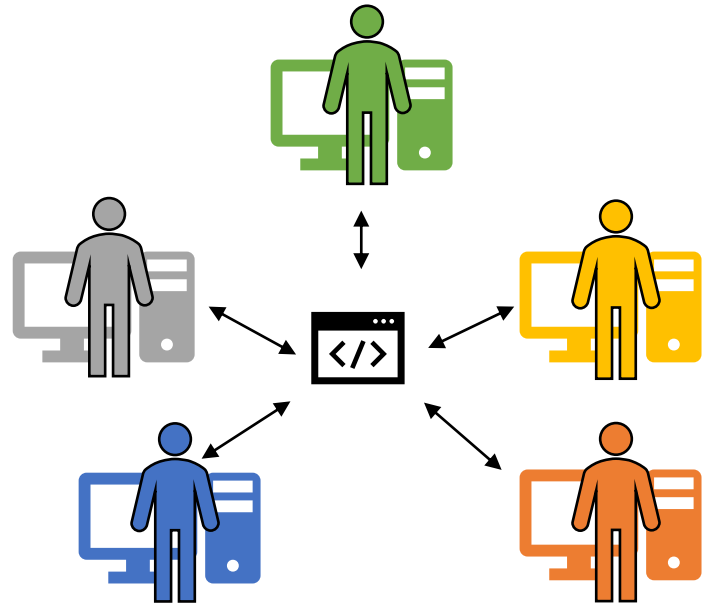
Properties of a permissioned ledger

- Byzantine Consensus Protocol
  - Safety
  - Valid when less than 1/3 replicas misbehave
- Ledger
  - Events recorded in a ledger
  - Omit transactions when 1/3 or more replicas misbehave
  - Rewrite the ledger when 2/3 or more replicas misbehave



Auditing permissioned ledgers

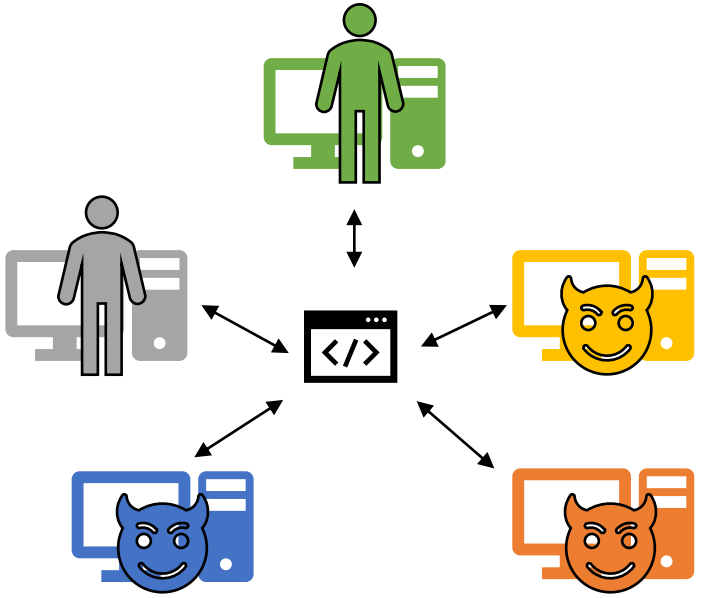
- Auditing detects when a problem occurred
- All replicas and their members are blamed for the misbehavior.

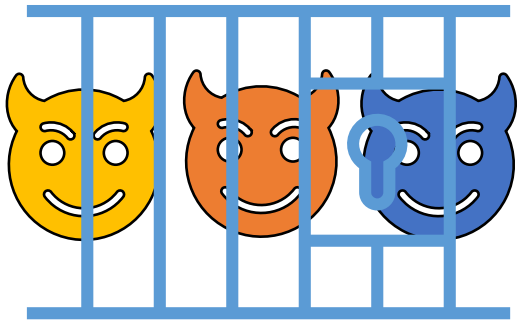


- **Goal**: Individual accountability

Individual Accountability

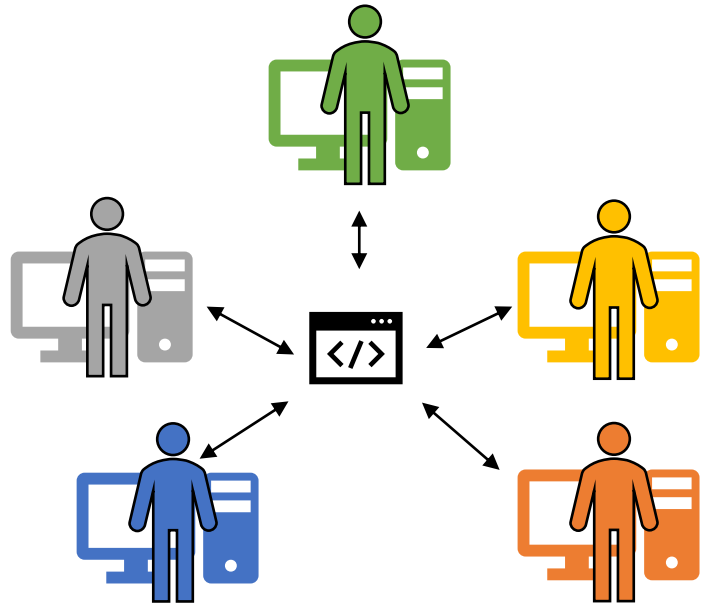
- Misbehaviour results in consequences
- Misbehaviour must always be detectable





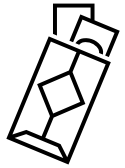
Co-designing the permissioned ledger

- **Co-design**
- Minimize signatures
  - One replica signature to commit a batch
  - Signature in receipts and ledger
- Receipts
  - Response in 2 round trips
  - Replicas commit to execution

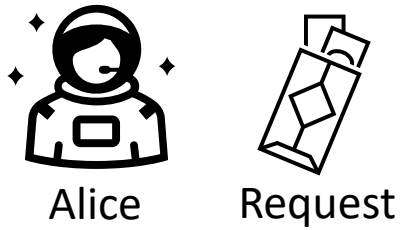


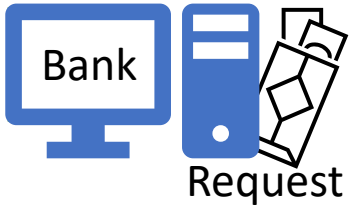
IA-CCF and L-PBFT


Alice


Request

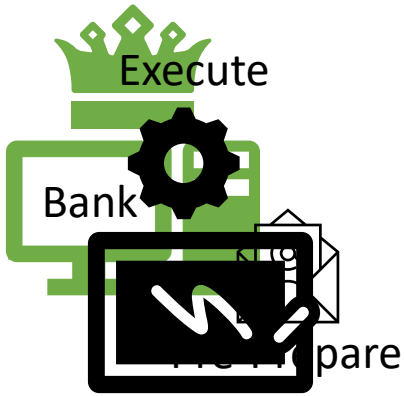
Sending a request





Early execution: Primary replica

Alice



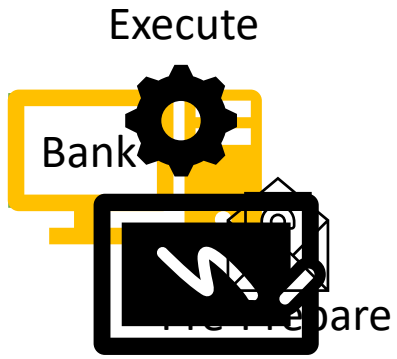
Write to Ledger

Pre-Prepare

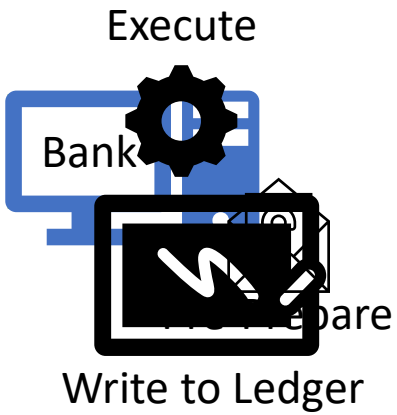
Pre-Prepare

Pre-Prepare

Early execution: Backup replicas

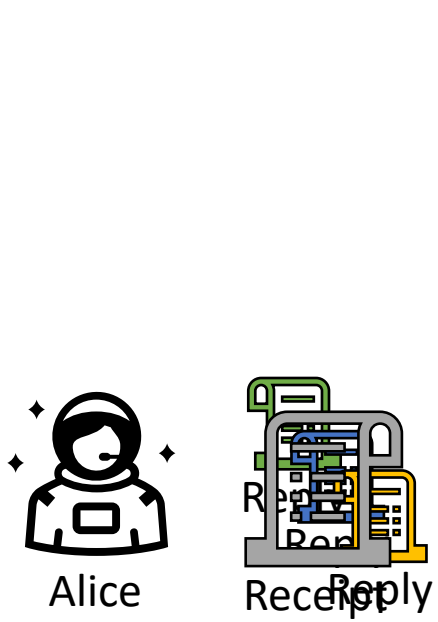



Alice



Agreement on execution

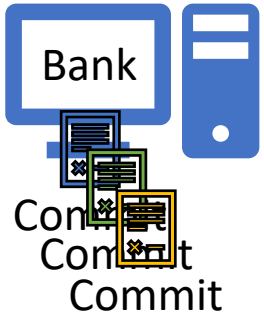
Creating a receipt



Committing the request

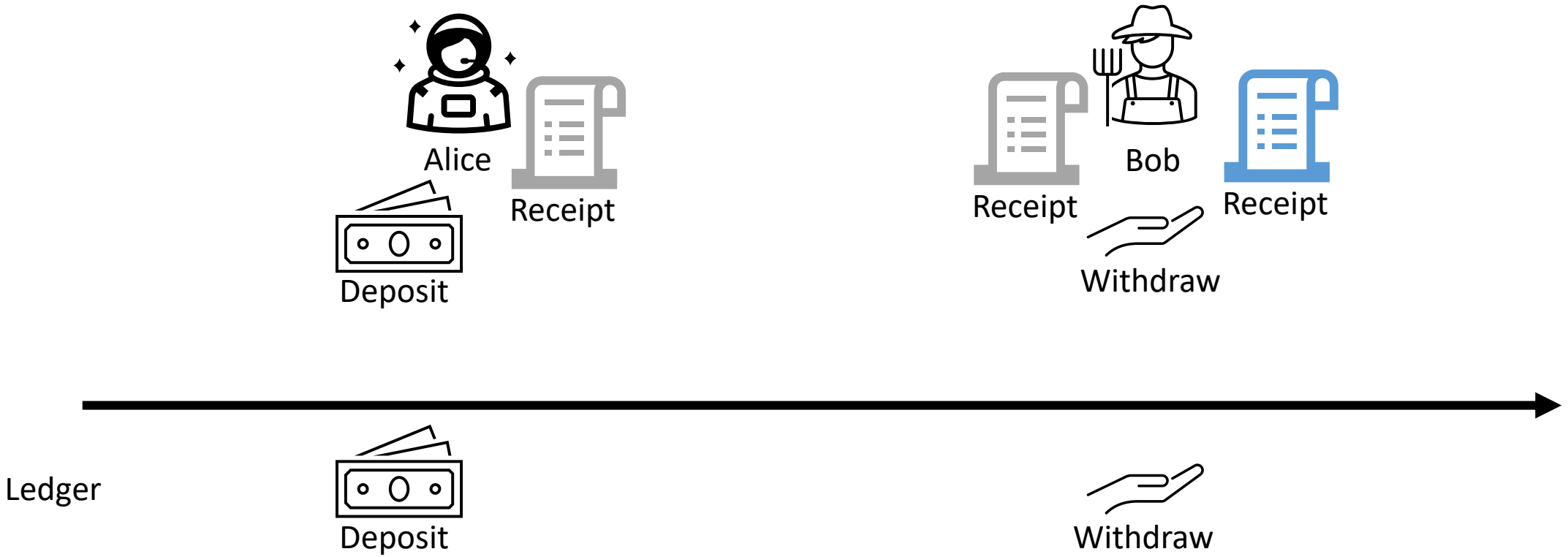
Alice

Receipt



Example: Misbehaving replicas

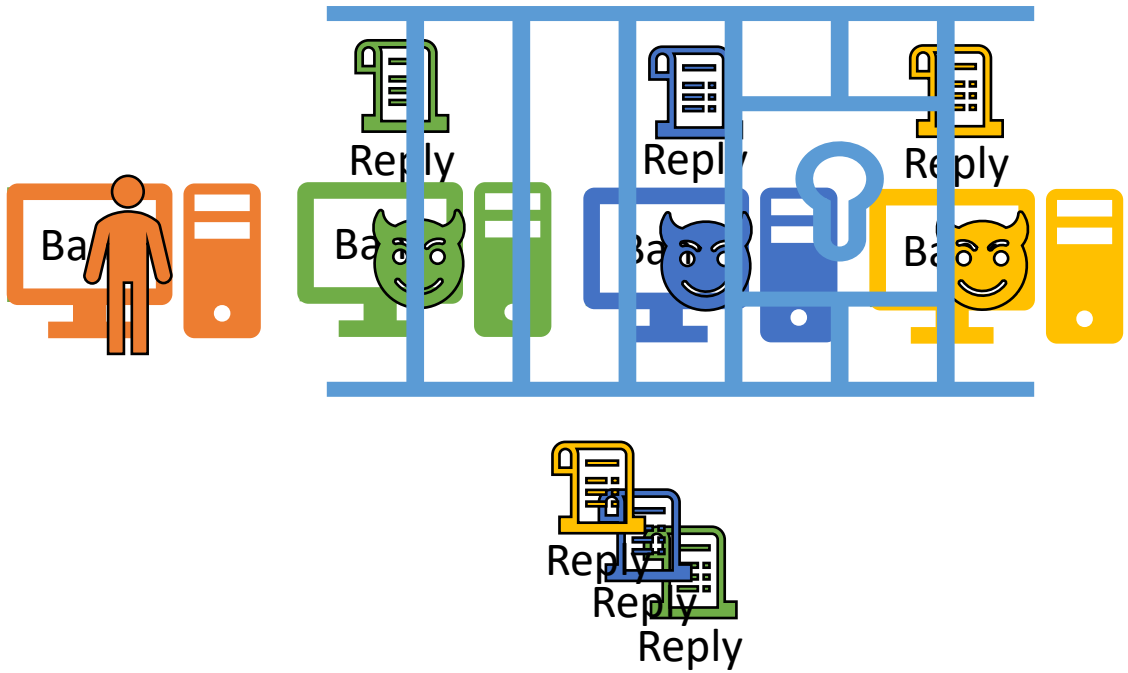
Example: Banking transactions

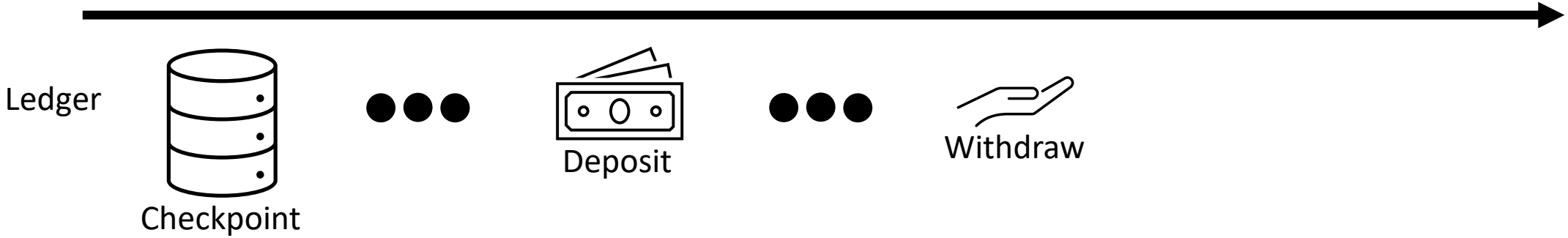


Auditing: Banking transactions

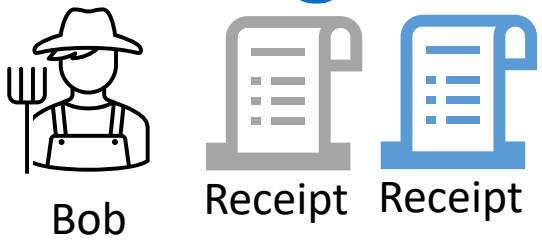
Bob

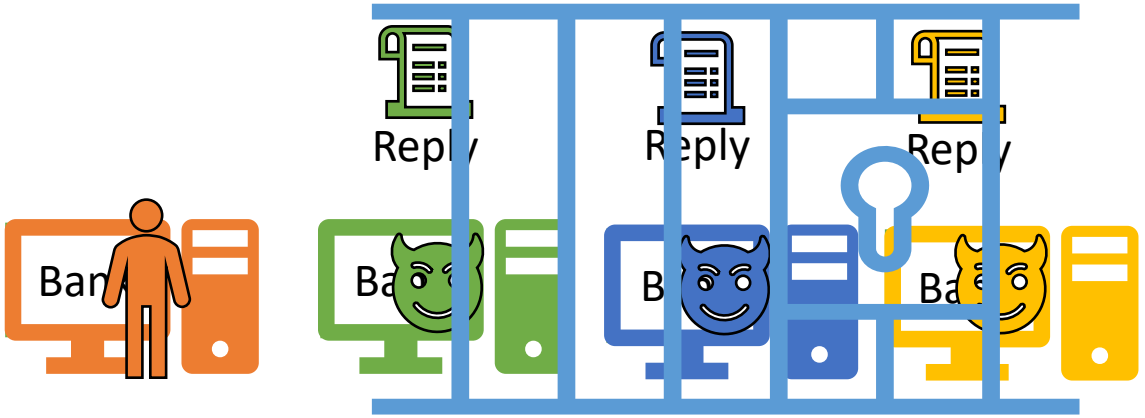
Receipt Receipt

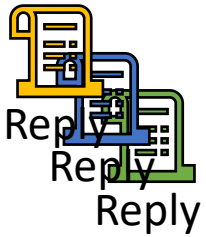


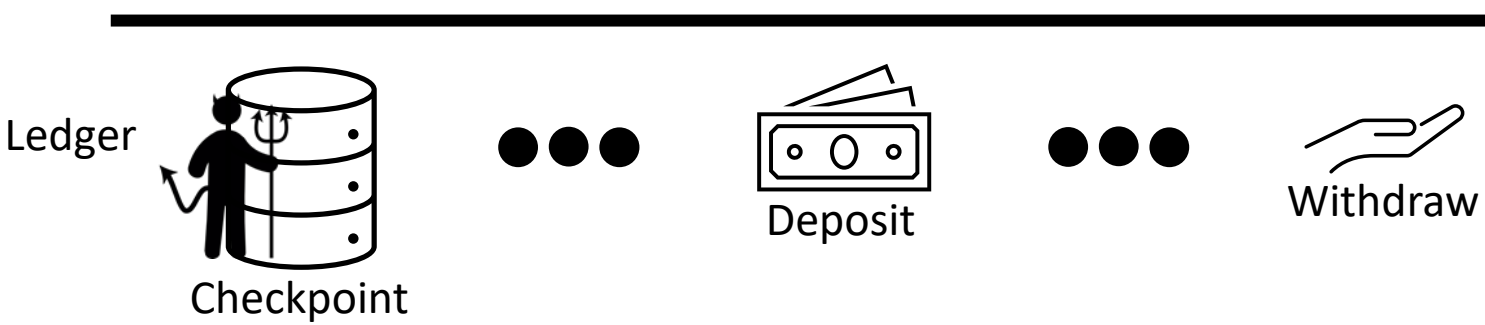


Auditing: Banking transaction







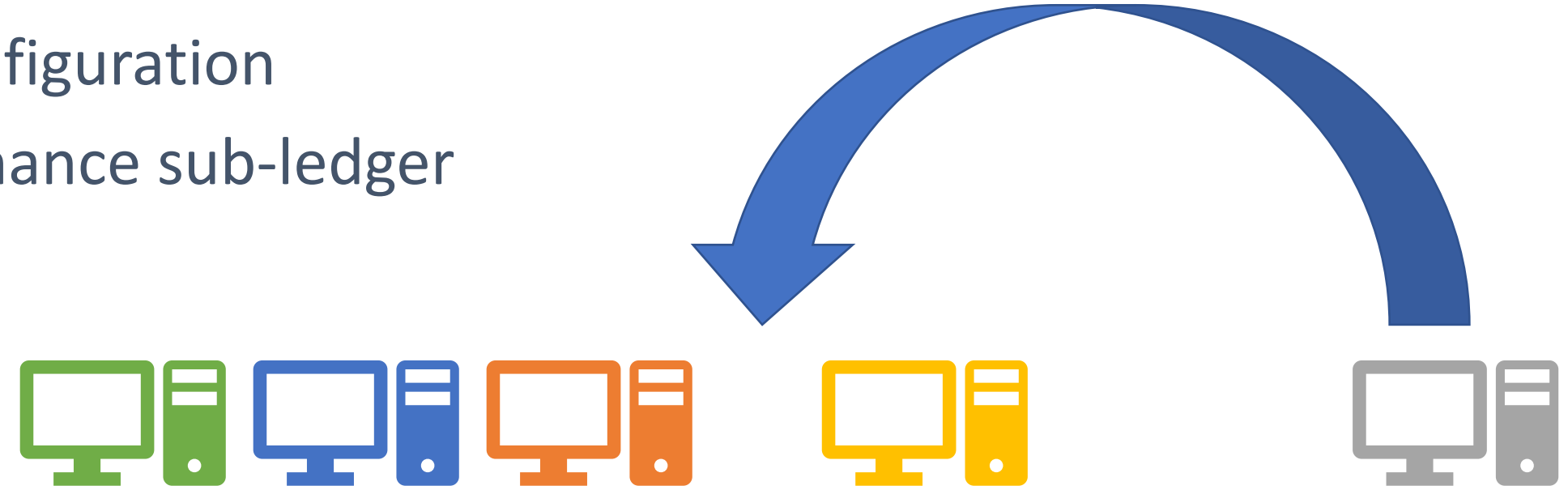


Auditing: Banking transaction

What if the malicious members also changed the replica set?

Changing the replica set

- Governance protocol
  - Member propose change
  - Members vote to accept change
  - Reconfiguration
- Governance sub-ledger



Receipts and the active replica set

- Receipts
  - Governance sub-ledger
  - Genesis transaction to current replica set

- Details and *proof* in the paper

Implementation

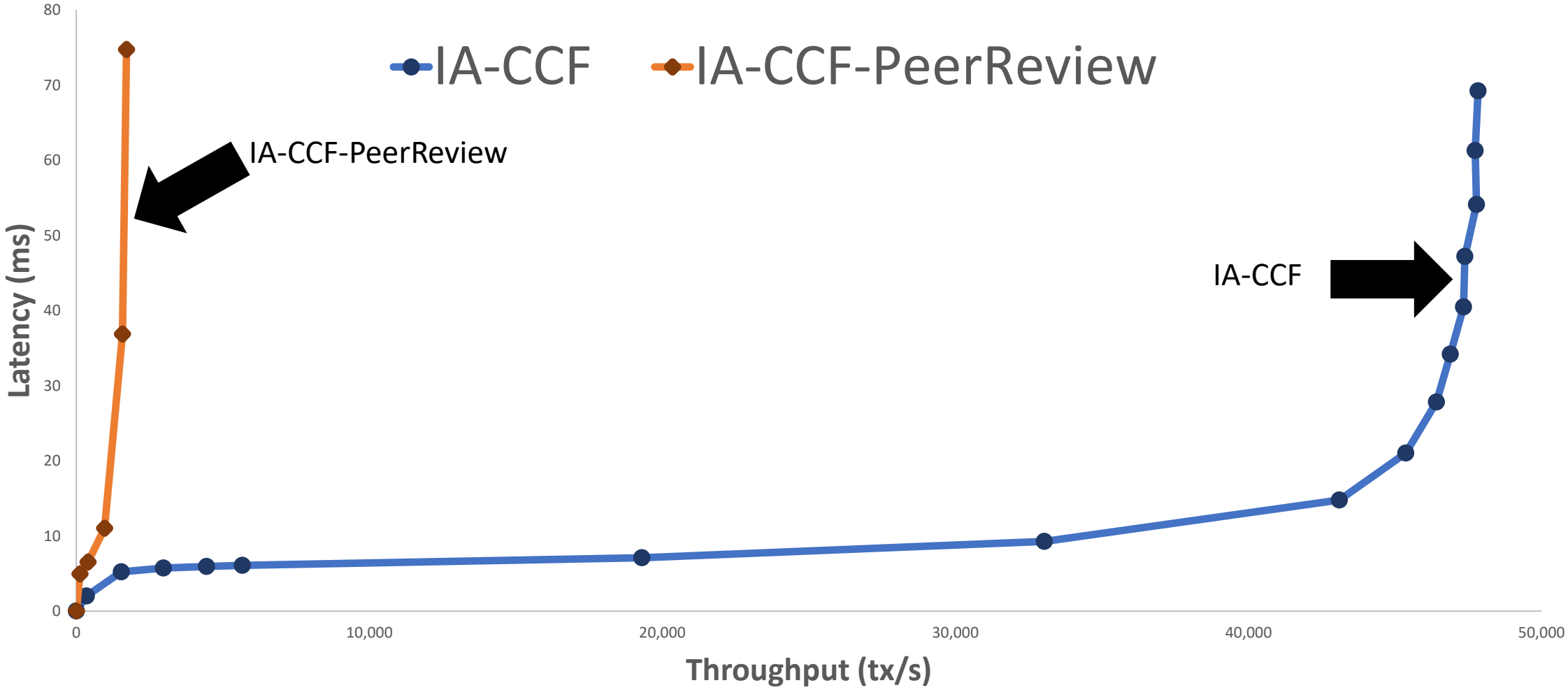
- Built IA-CCF on top of *CCF:*
  - Transaction engine and key-value store
  - Ledger and receipts
  - Programmable Governance protocol
- IA-CCF co-designed:
  - Byzantine consensus protocol (L-PBFT), Ledger, Receipts, and Governance protocol

http://ccf.dev

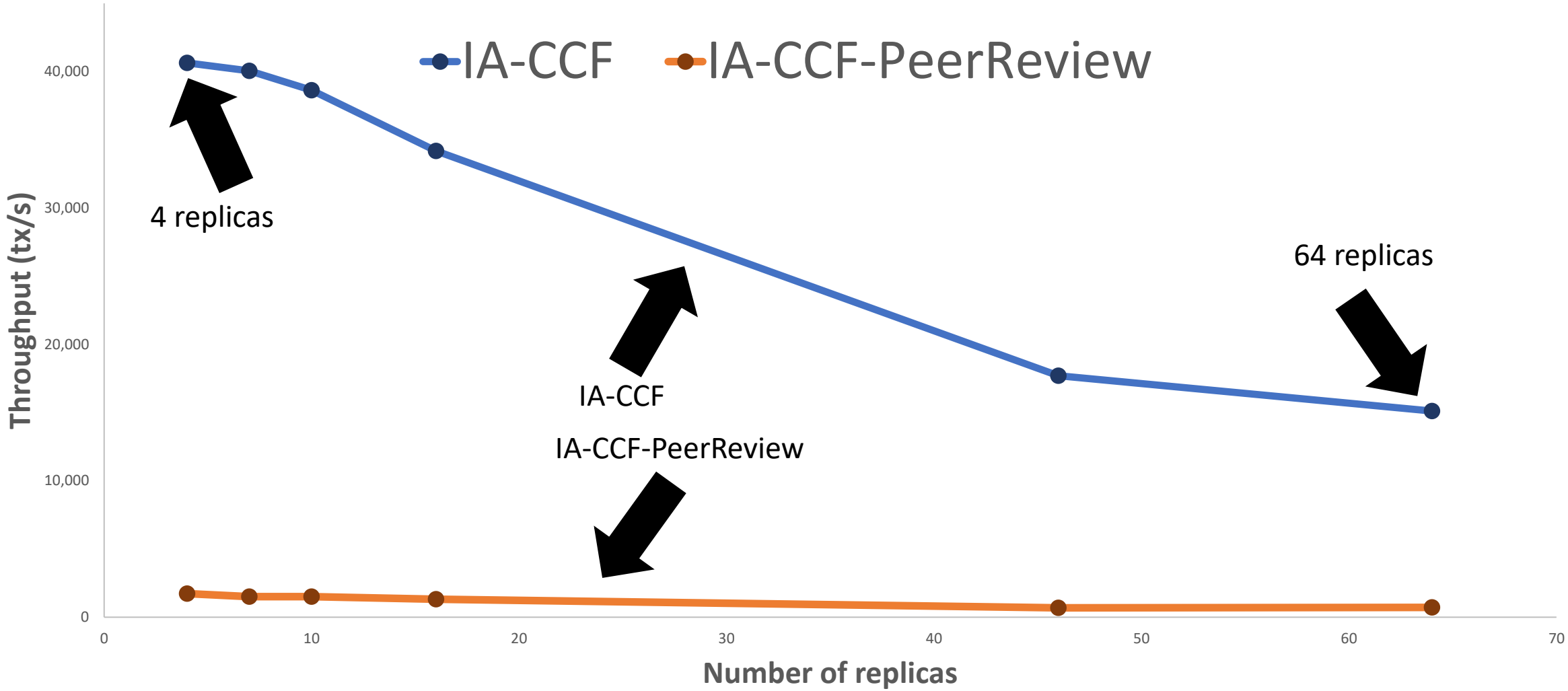
Performance evaluation

- Smallbank benchmark
  - Transfer funds between accounts
- LAN environment
  - 1 datacenter
- WAN environment
  - 3 geo-distributed Azure datacenters

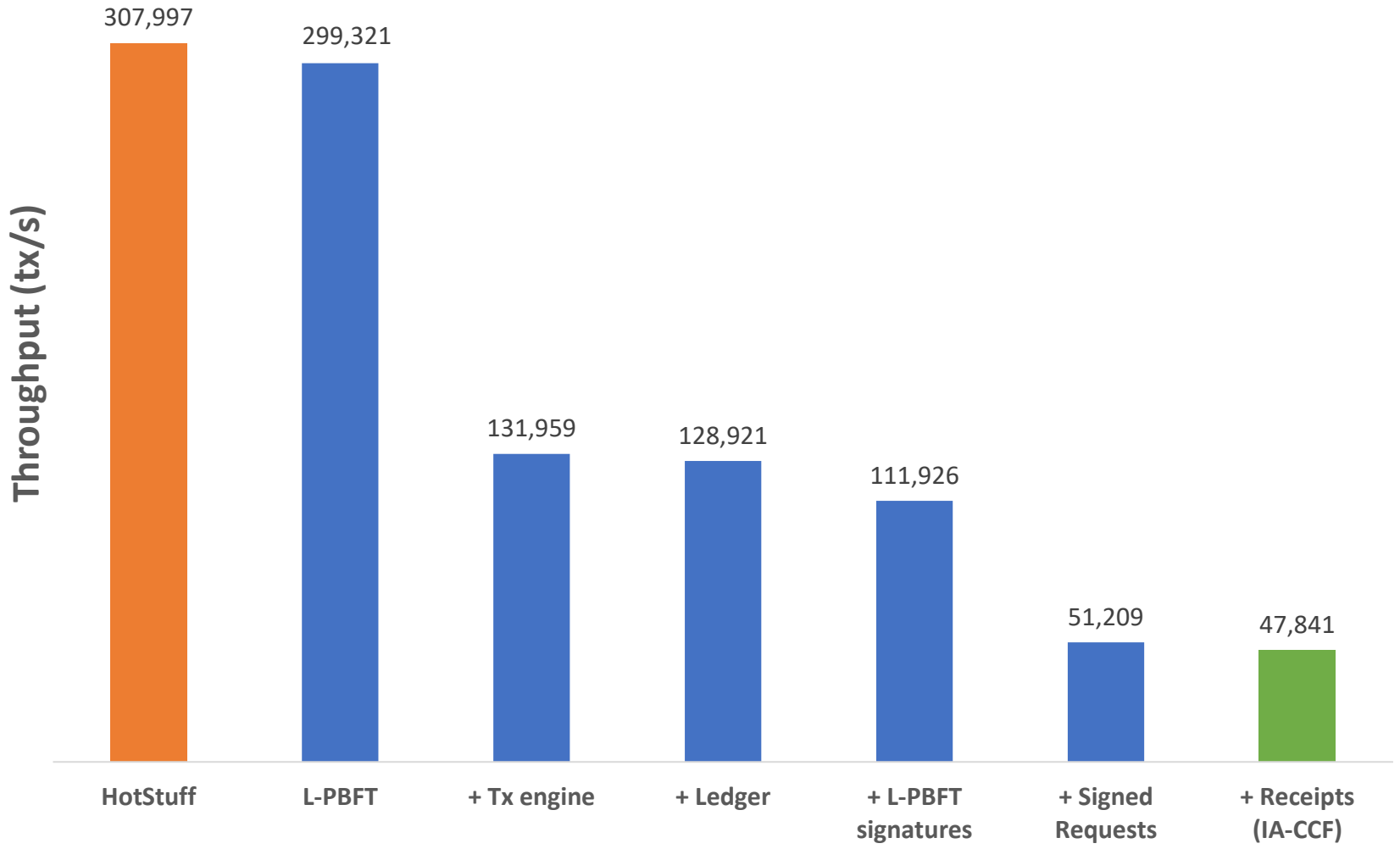
Throughput vs Latency (LAN)



Scale-out (WAN)



Performance of IA-CCF features



Conclusion

- Individual Accountability
  - Only blame replicas that misbehave
- Co-design - Byzantine consensus protocol, Ledger, Receipts, and Governance
  - 1 signature per batch
  - Receipts returned to client in 2 round trips
- Auditing with individual accountability proof

http://ccf.dev