=nil; Foundation

ShardDAG

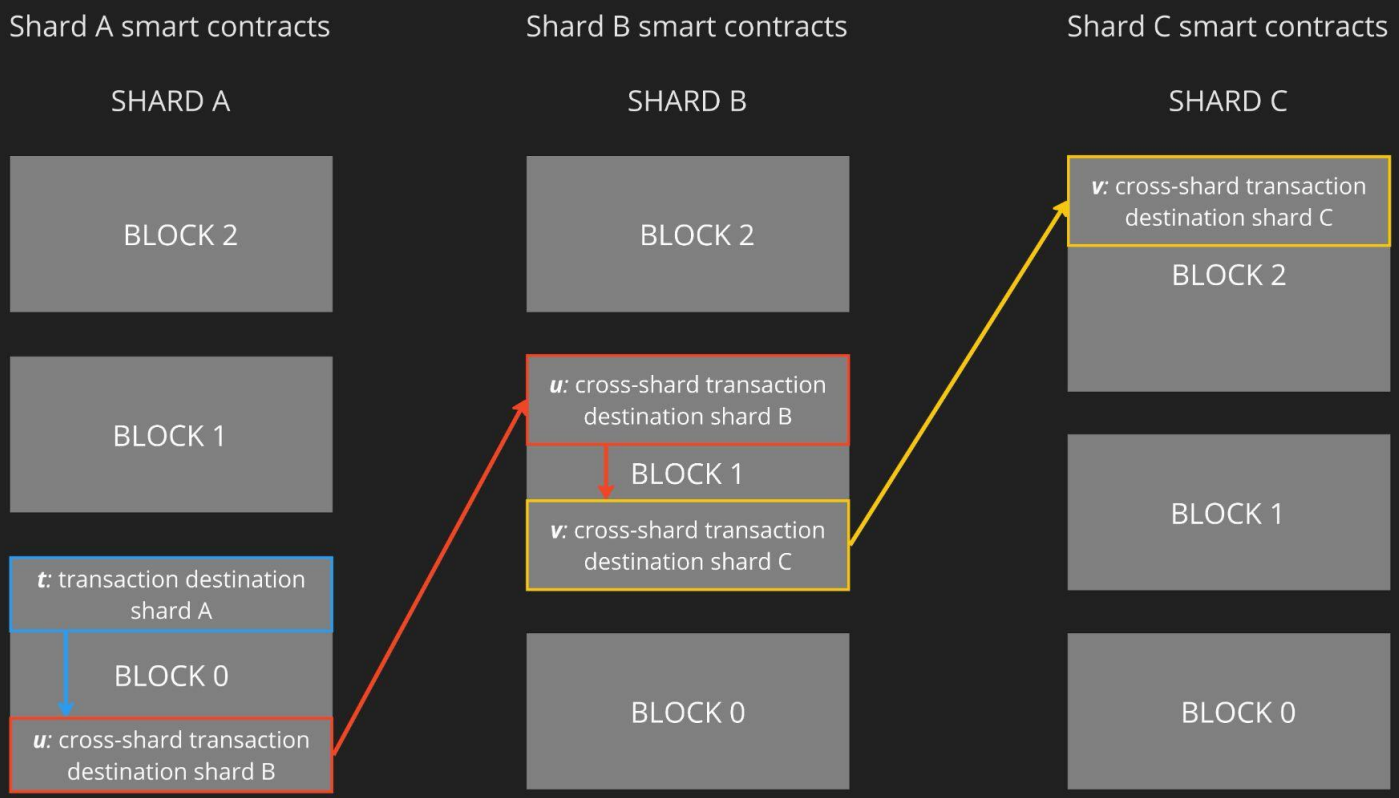
Ordering a Sharded Blockchain

James Henderson

=nil; Tech Talk

zkSharding Review

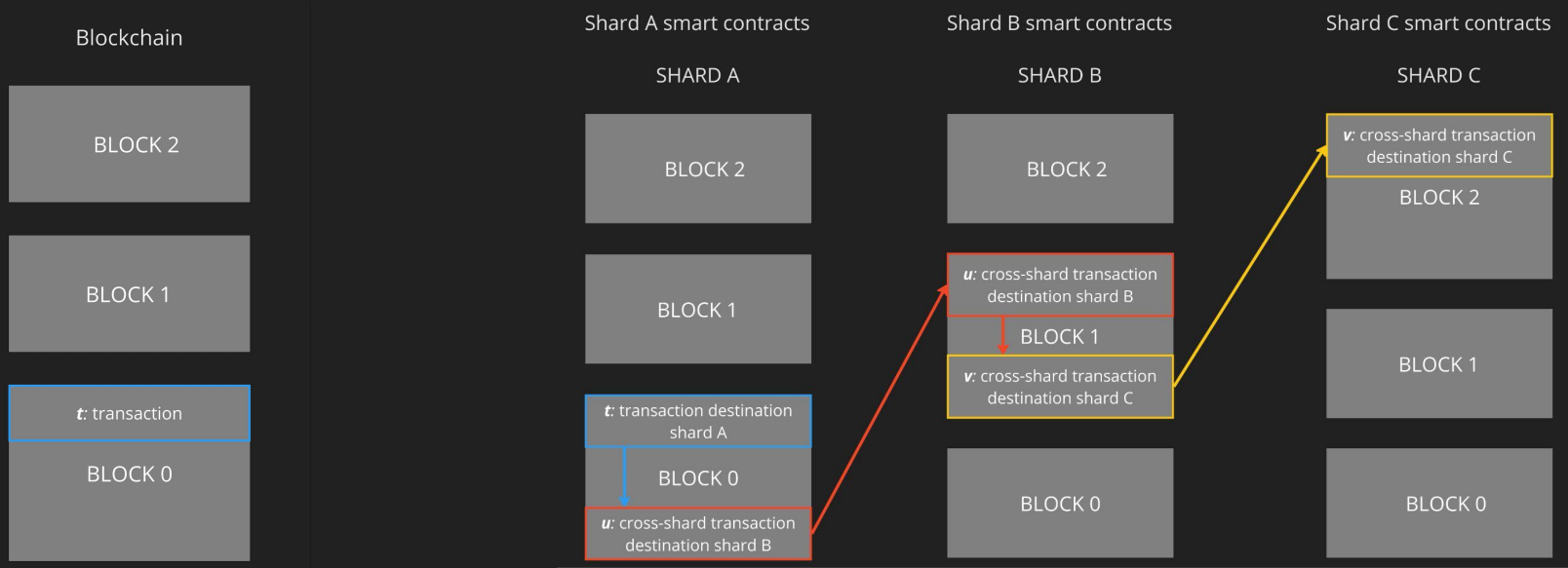
State Sharding and Cross-Shard Transactions



Exploits, Censorship, Regulation, Fairness, Decentralisation … Getting Data On-Chain

- Blockchain problem: Data on-chain is secure, but getting data on-chain is (potentially) dangerous, problematic
  - Centralising force - PBS potential solution
- Exploits: sandwiching, frontrunning, censorship, regulation, oracle manipulation, ...
- If money can be made from an exploit, somebody will do it, potentially spawning an entire industry
  - High frequency trading, traditional finance
- Blockchain is still new - exploits will continue to be developed

Sharding Magnifies Exploitation Problems



Non-Sharded:
Transaction completion requires cooperation from a single block proposer

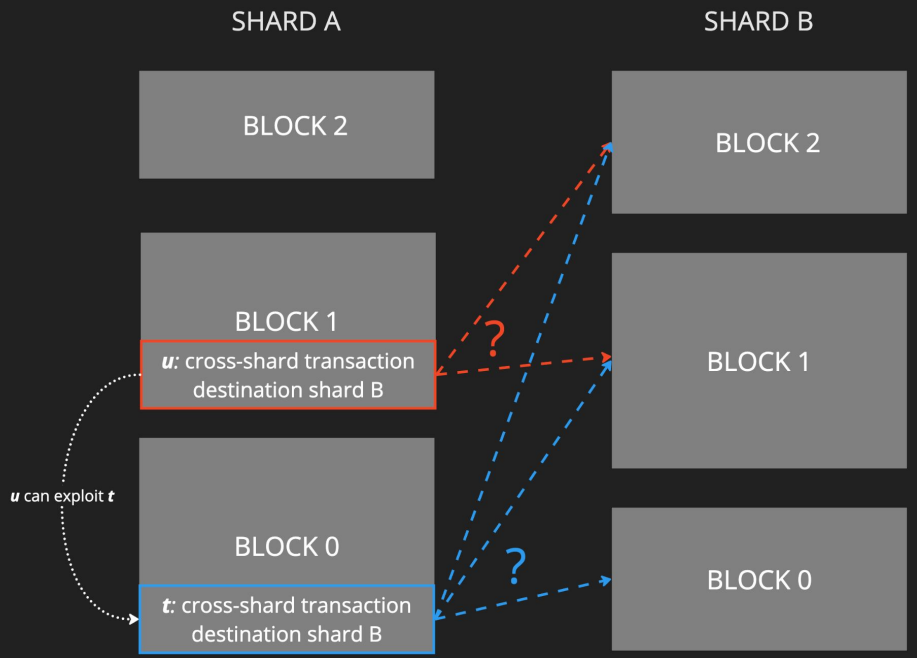
Sharded:
Transaction completion may require cooperation from many shard block proposers

Cross-Shard Transactions & Ordering

Enforce ordering to constrain exploits

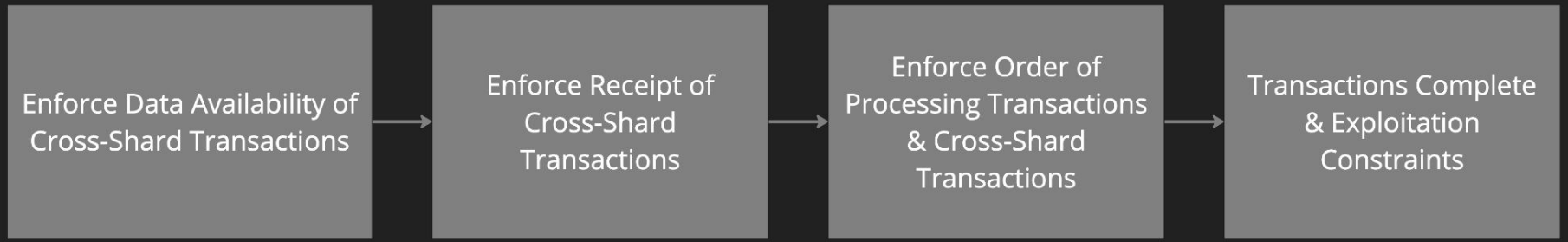
- Punishment for not processing a cross-shard transaction, or processing in an incorrect order.
- Order is enforceable if it can be established that all the required data was available to the shard and that the shard subsequently failed to process it correctly.
- A mechanism is required for establishing cross-shard transaction data availability.

- TON has no verifiable, enforceable ordering
  - exploitable



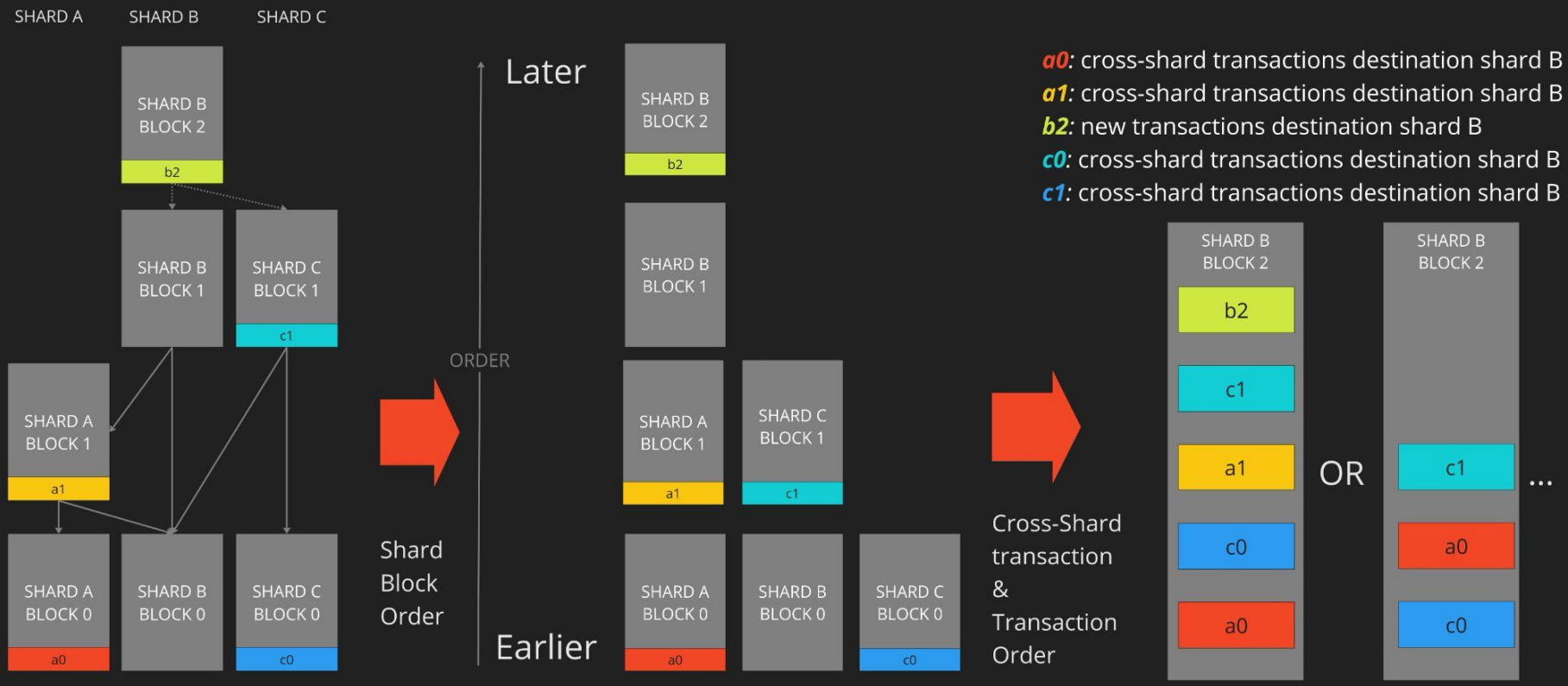
Strategy

zkSharding employs a shardDAG architecture that combines protocol rules, rewards and penalties to constrain the ability of validators to manipulate the processing of transactions.



ShardDAG

Directed Acyclic Graph - Partial Order



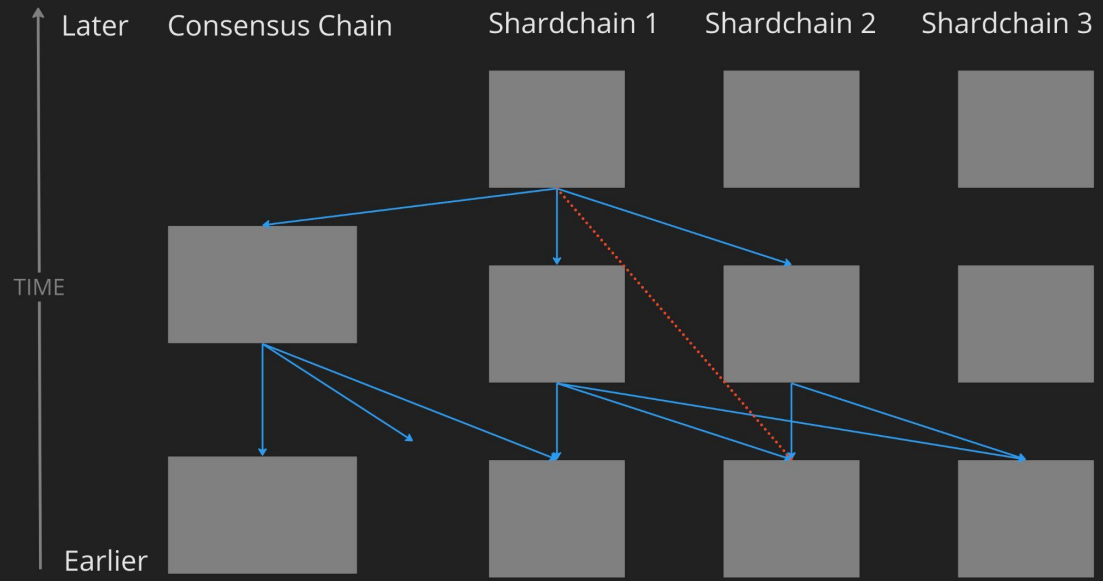
a0: cross-shard transactions destination shard B
a1: cross-shard transactions destination shard B
b2: new transactions destination shard B
c0: cross-shard transactions destination shard B
c1: cross-shard transactions destination shard B

Shard Block Data Structure

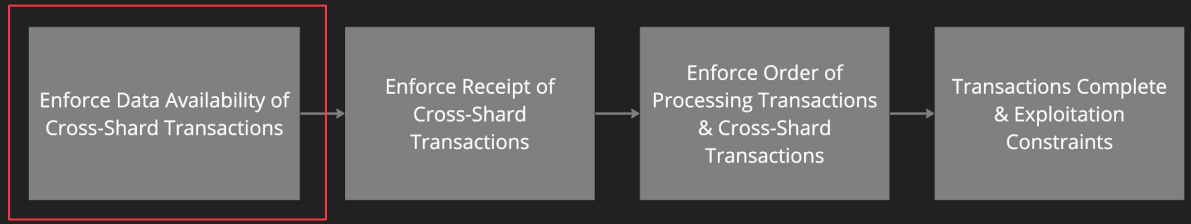
Shard Block Header Contains:

- Hash to previous block
- Hashes to other shard blocks
- Hash to consensus block

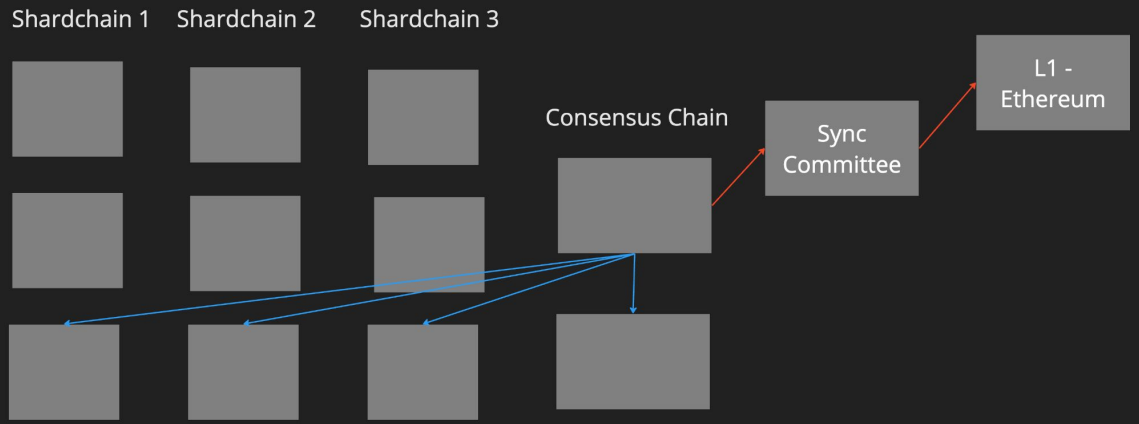
Hashes acknowledge receipt of shard block header and outbox of cross-shard transactions, *for entire subgraph.*



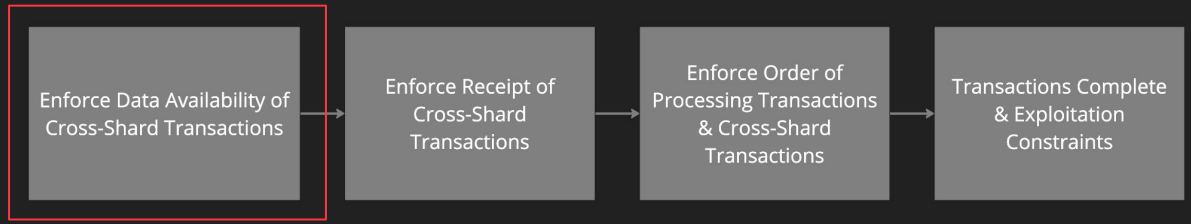
Consensus Chain



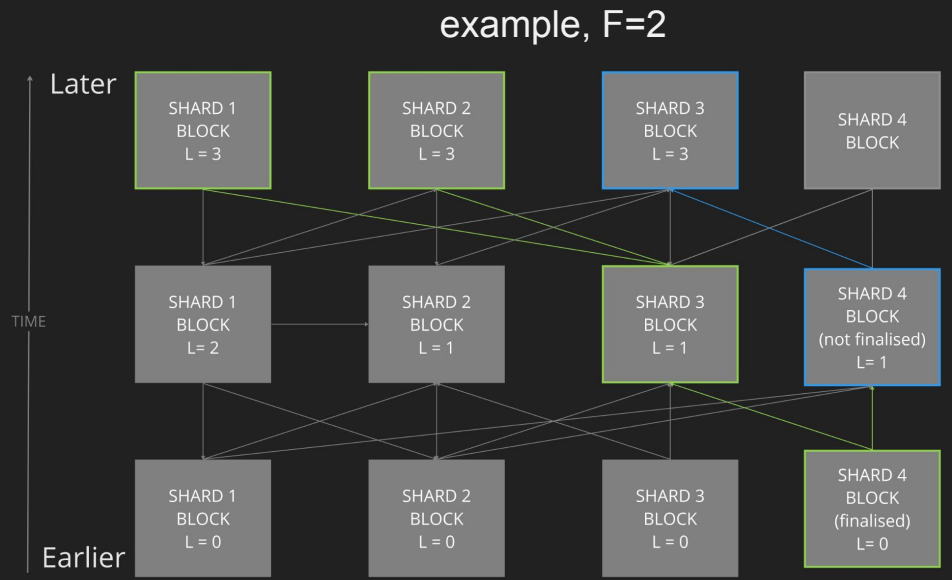
- Consensus chain checks for some malicious shard behaviour
- Shard blocks must be included in consensus chain
  - To reach finality
  - For validators to get paid



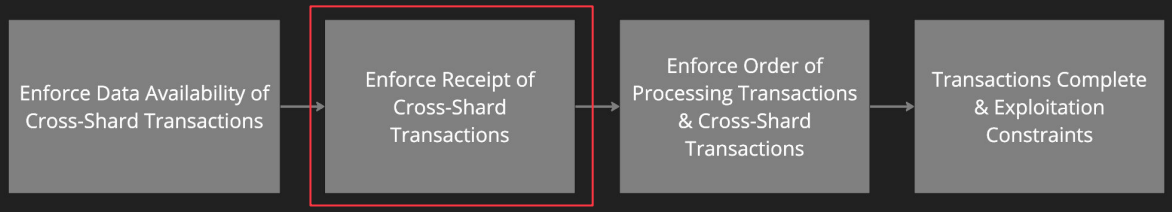
Sending Cross-Shard Transactions



- F is a parameter controlling DAG branching
  - Controls rate of data ‘spreading’ amongst shards
- [CHILD CONDITION]: For a shard block *b* to be finalised,
  - *b* received by >F shards (via hashes) within the consensus chain subgraph

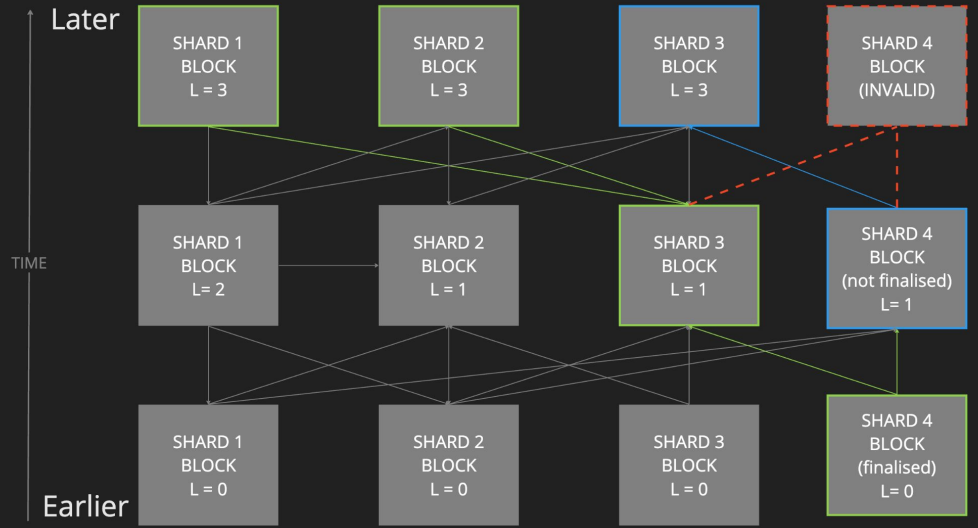


Receipt of Cross-Shard Transactions

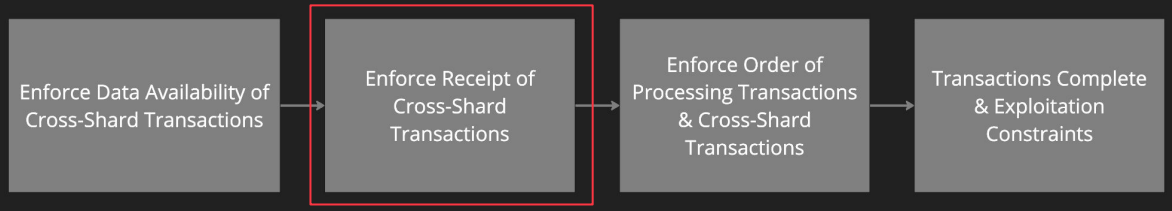


- [PARENT CONDITION]: For shard block *b* to be valid
  - *b* must acknowledge receipt of *new* shard blocks from >F shards
- Shards/validators must receive shard block data to get paid

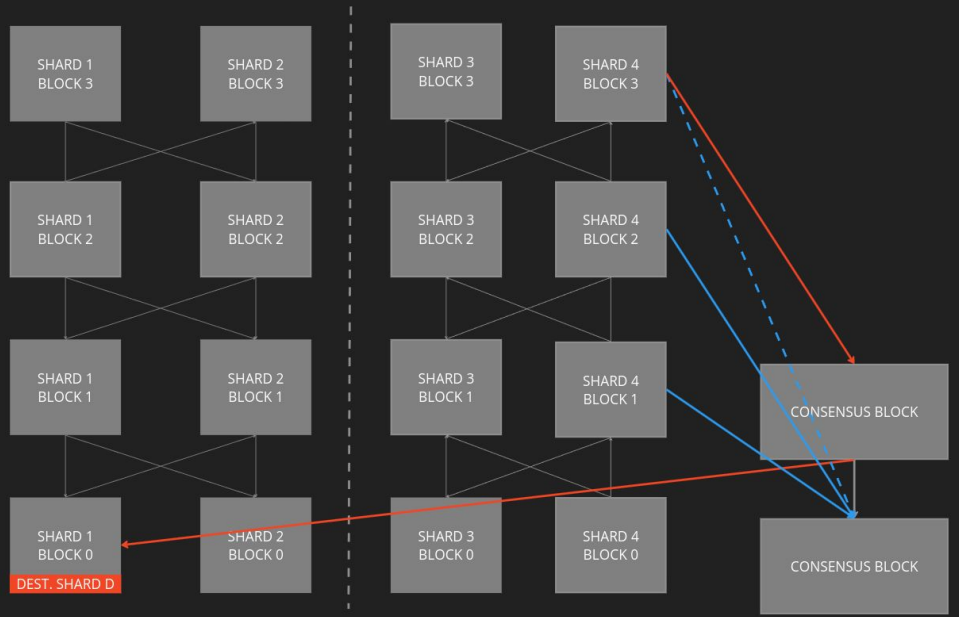
example, F=2



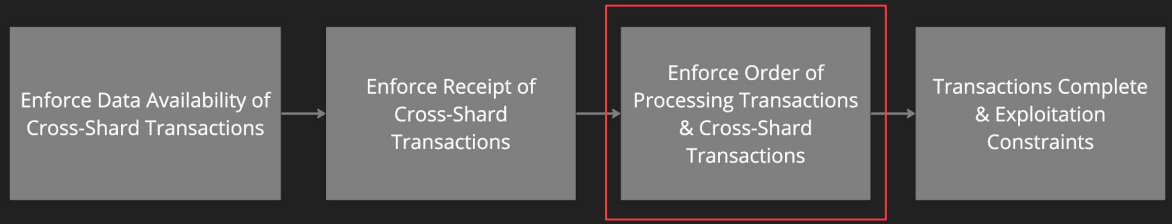
Receipt of Cross-Shard Transactions



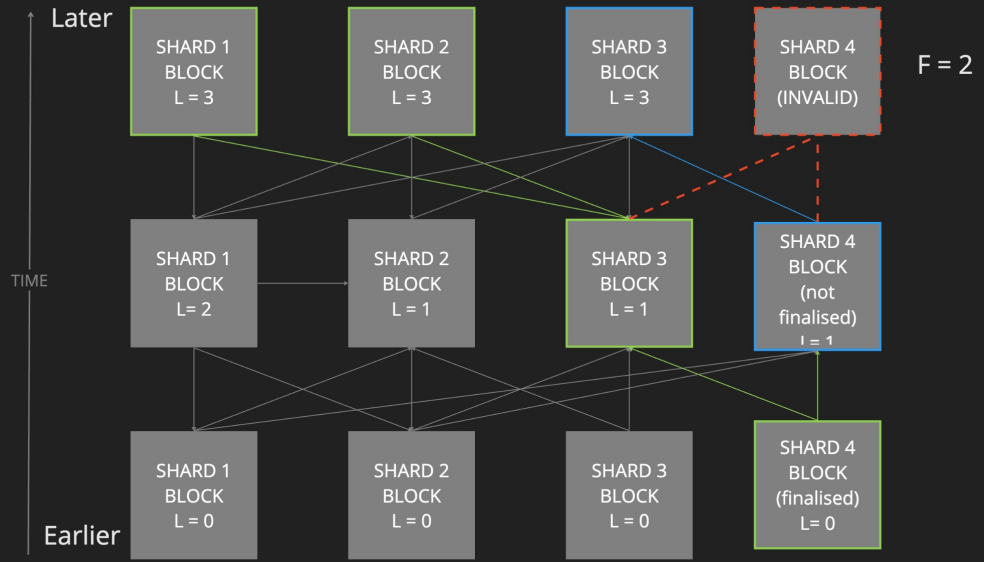
- [CONSENSUS PARENT CONDITION]:
  - Only *X* consecutive shard blocks can connect to the same consensus block
- Guarantees cross-shard transactions (eventually) reach destination
- Shards/validators must receive shard block data to get reward



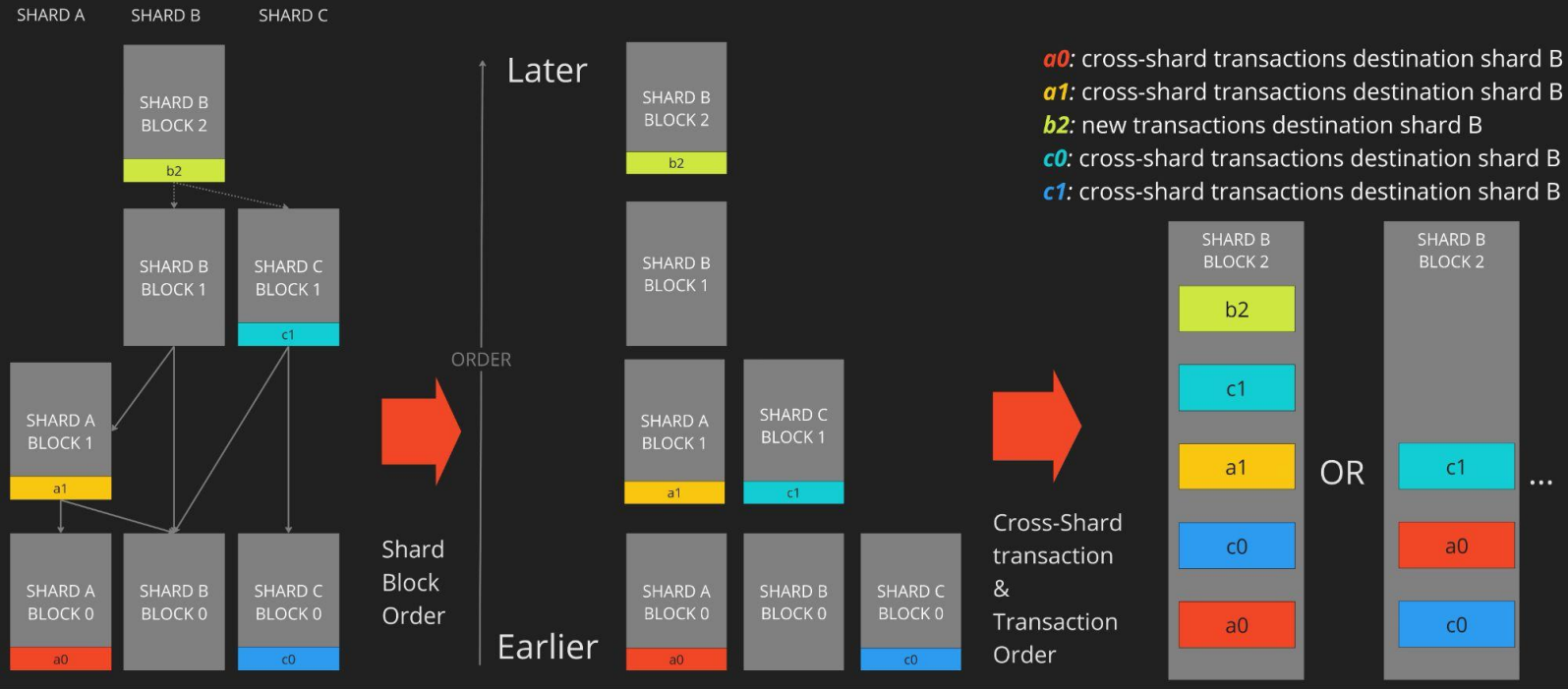
Ordering Cross-Shard Transactions



- Lamport Timestamps (logical clock)
  - L : 1 more than largest parent
- Processing order within a block's subgraph:
  - L=0,
  - then L=1,
  - then L=2,
  - … until block is full



Ordering Cross-Shard Transactions

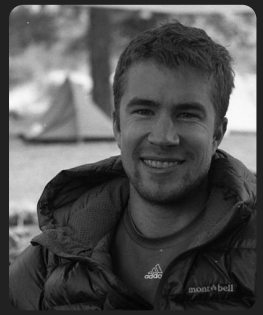


- Cross-shard transactions inherit the ordering of shard blocks from the DAG
- Processing anything outside a shard block's subgraph is invalid.
- Retain order of multiple CSTs in a single shard block.

For the Future

- Data Storage
  - Deleting processed cross-shard transactions that are no longer needed
  - Sampling
- Communication overheads
  - Efficient messaging in the P2P network
  - Push and pull messaging
- Other ordering criteria
  - Fees
  - Async locks
  - Overloaded shards

Thank you!



Questions?

Notion: [Shard Interaction Architecture [current]]