Lecture 12: Layer 2 Scaling: Side Blockchains

https://web3.princeton.edu/principles-of-blockchains/

Professor Pramod Viswanath
Princeton University

This lecture:
Layer 2 Scaling -- no need to change the consensus;
Side chains; data availability, outsourcing execution and storage

Trust from Trusted Blockchain

- Trusted Blockchains
  - Bitcoin, Ethereum

Safe and live for a long time

Poor performance

- New Blockchains
  - Ouroboros, Prism

High performance

Wholesale change

- ***Side Blockchains***

Same layer 1 structure

Simple, efficient and practical

  - Derive trust from trusted blockchains
  - Commit hashes periodically

Nakamoto's solution:
Simple Payment Verification

Lite node

SPV node

Only wants to verify a transaction

Download chain of block headers from main node

header: nonce, parent.hash, Merkle root

periodically poll main nodes

update if there is a longer chain

Layer 2 Scaling

Simple payment verification

specific to Bitcoin (UTXO state structure)

no need to change consensus (layer 1)

Need to generalize this solution

should be able to verify state (this lecture)

provide a platform of its own to blockchain participants

- trust is derived from the layer 1
- layer 2 overlay for efficiency
- focus of next two lectures

Limitations of Light Node Verification

Lite node

needs to be connected with a majority of honest nodes

This is a strong (networking) requirement

easily attacked via botnets

Verification via Fraud Proofs

Interactive verification mechanism

needs lite nodes to be connected to at least one honest node

Security depends on fraud proofs

fraudulent activity is detected by full nodes that can furnish proof of fraud

proof of non-inclusivity in the Merkle tree

Data Availability Attack

A malicious block producer publishes a block header so that:

1. light nodes can check transaction inclusion, but

2. withholds a portion of the block (e.g., invalid transactions), so that it is impossible for honest full nodes to validate the block and generate the fraud proof.

Conundrum for Honest Nodes

Honest full nodes are aware of the data unavailability
but there is no good way to prove it.

1. Best is to raise an alarm without a proof
this is problematic because the malicious block producer can release the hidden parts **after** hearing the alarm.

2. Due to network latency, other nodes may receive the missing part before receiving the alarm
thus, cannot distinguish who is prevaricating.

Need for Data Availability

For fraud proofs to work:

light nodes must determine data availability by themselves.

Key question:

when a light node receives the header of some block, how can it verify that the content of that block is available to the network by downloading the least possible portion of the block?

Data Availability as a Service: ACeD

scales data storage

Need to Encode the Block

A transaction is much smaller than a block

so, a malicious block producer only needs to hide a very small portion of a block.

Only way to detect such hiding by

the entire block is downloaded.

Encoding:

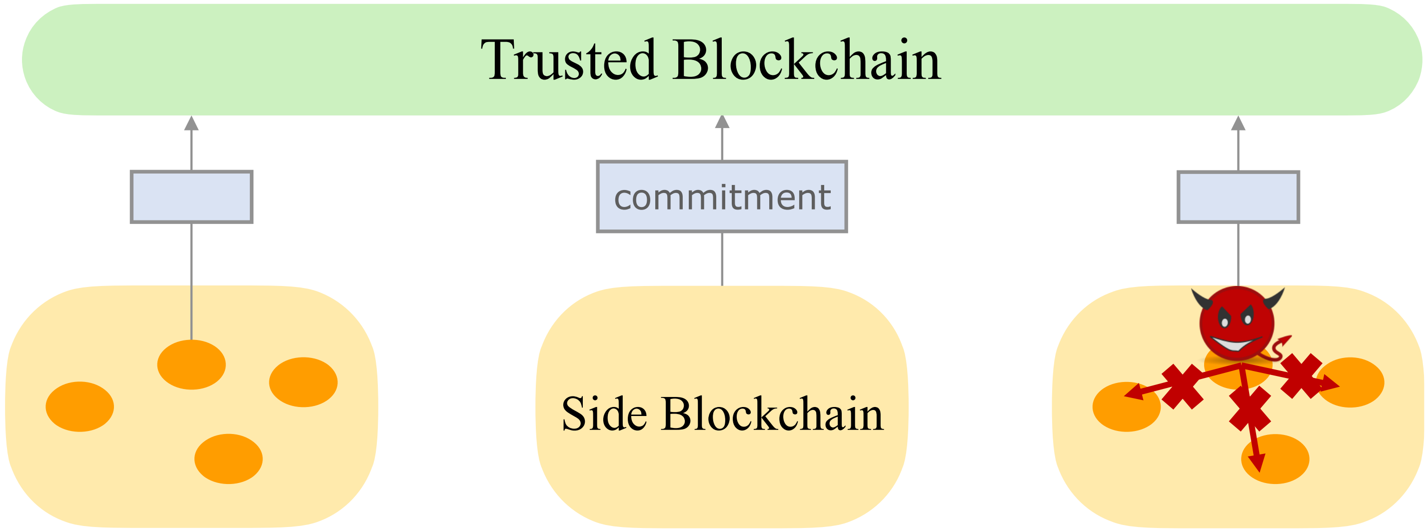
linear erasure codes

any small hiding will result in lot of data being unavailable

thus detected via random sampling

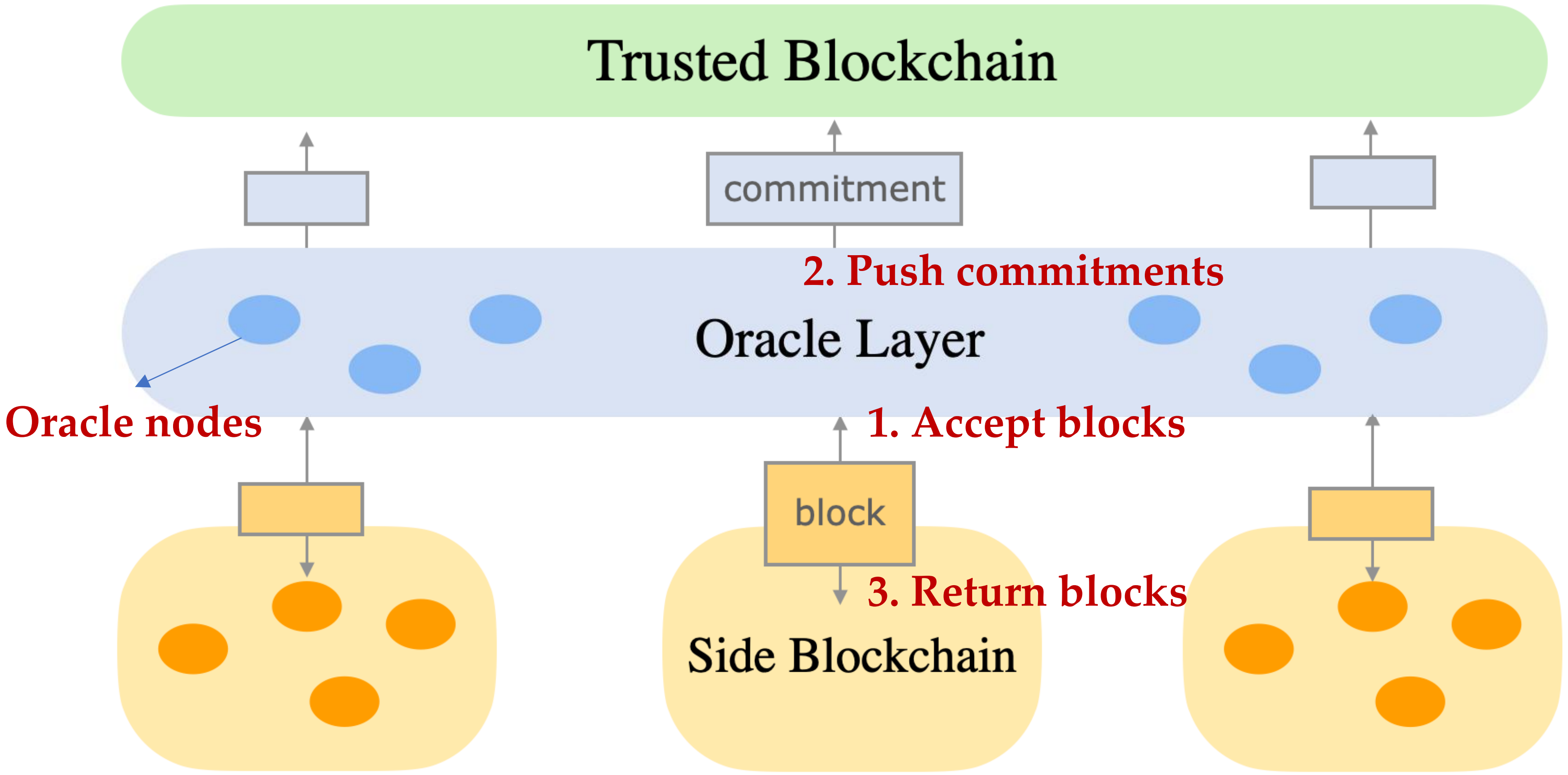
ACeD: Scalable Data Availability Oracle

Data Availability Attack

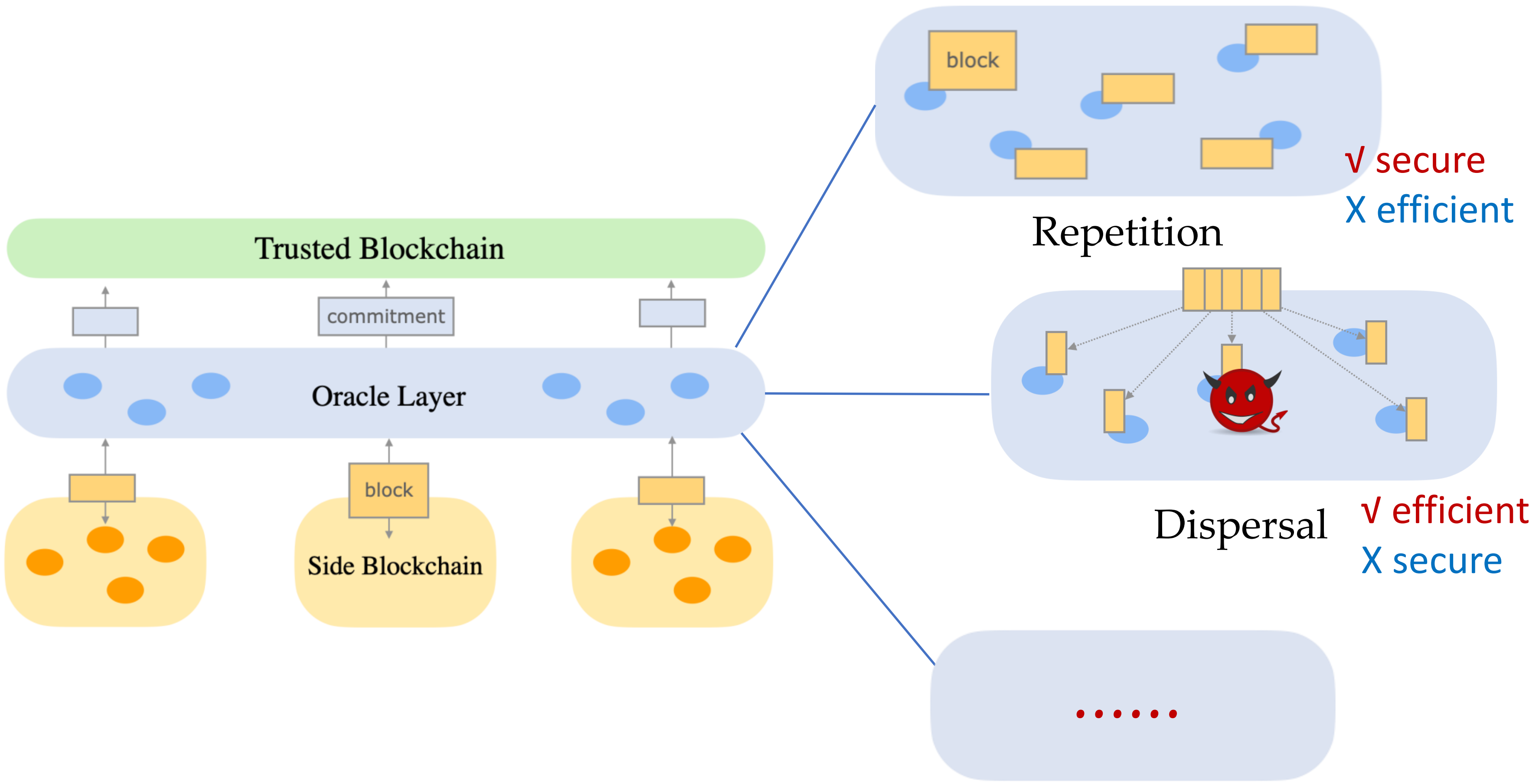


- Side Blockchains
  - Order of blocks ← order of hashes
  - *Attack: not transmit the block to others*

Data Availability Oracle

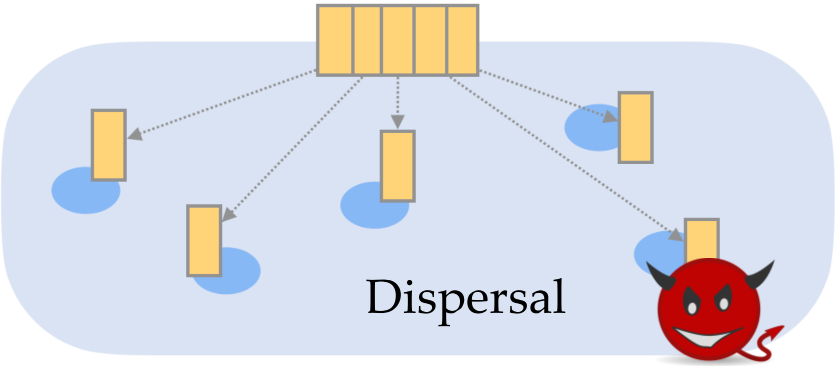


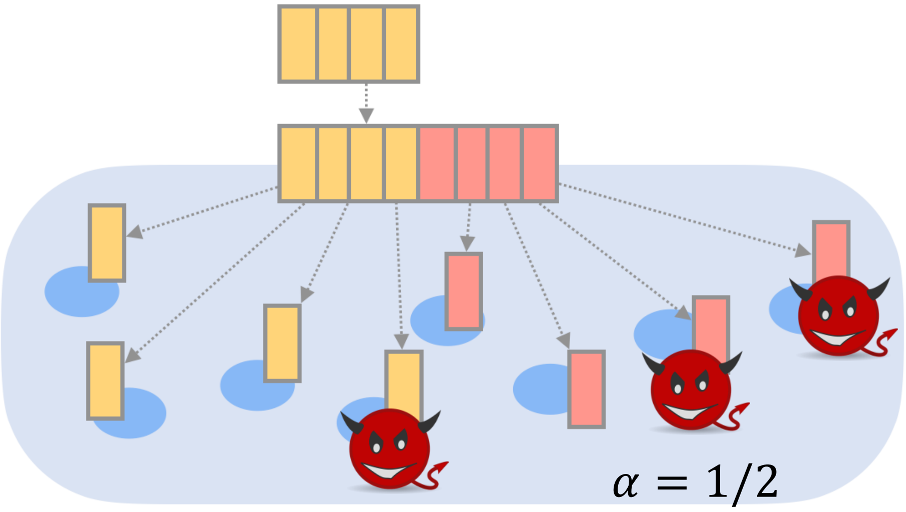
Repetition and Dispersal



Securely and efficiently

Coded Dispersal





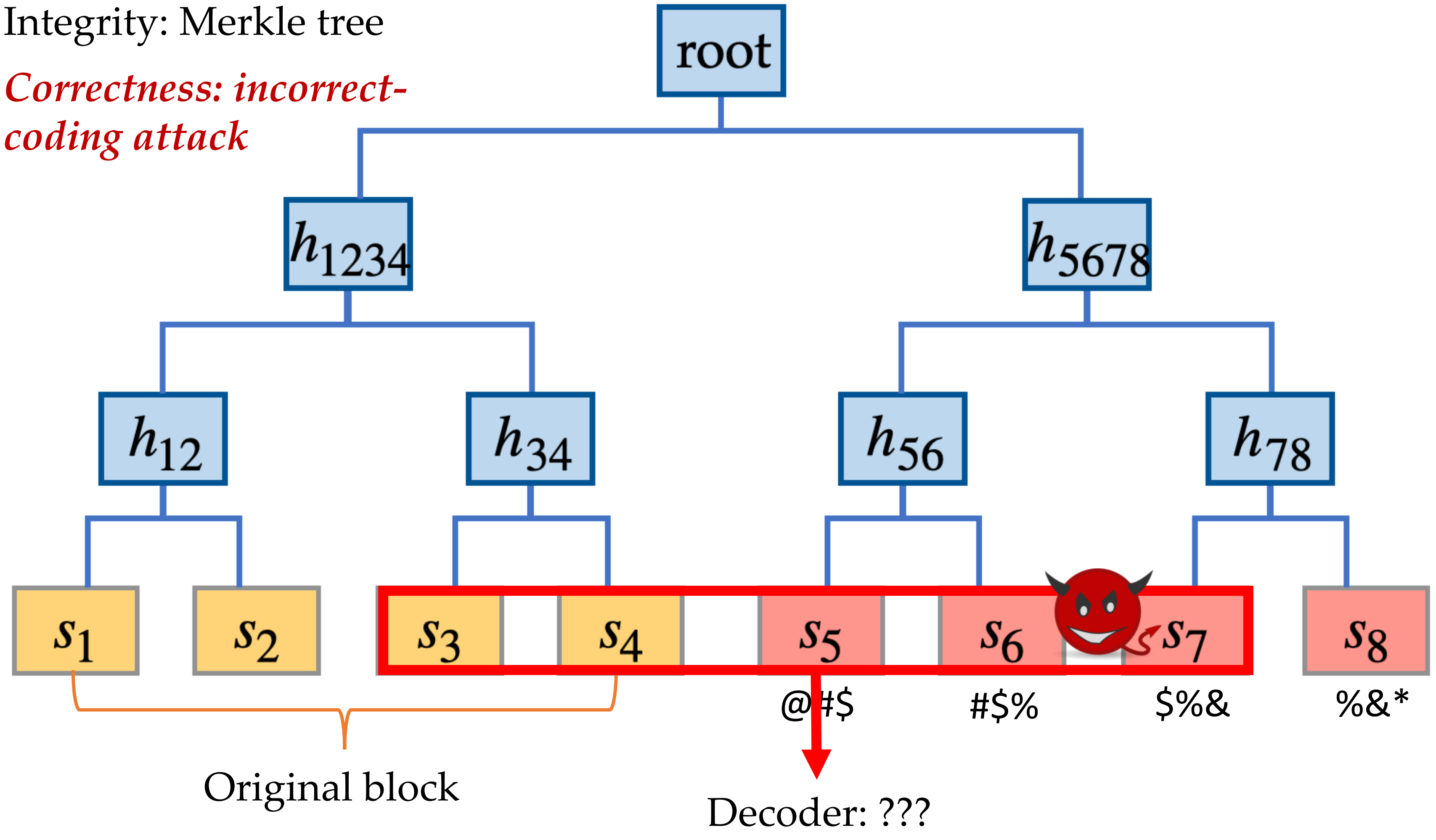
Coded Dispersal

- Erasure Coding
  - *(n, k) Reed-Solomon code*
  - k data symbols → n-k parity symbols (n coded symbols)
  - *Undecodable ratio* α
- Integrity and Correctness

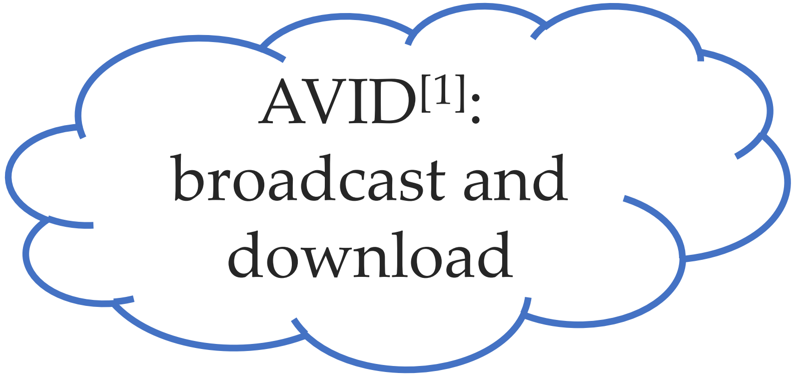
Incorrect-coding Attack

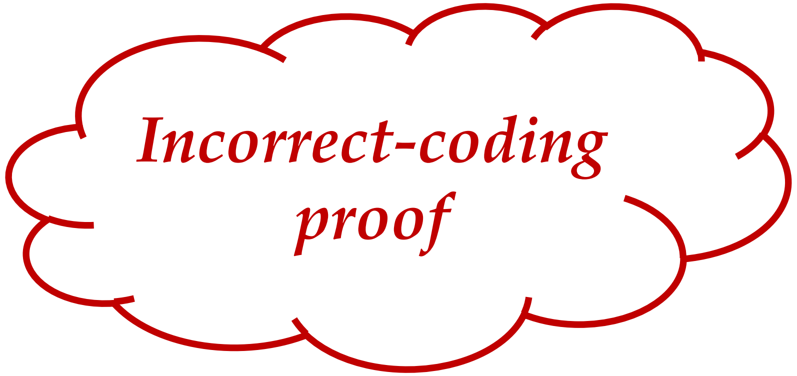
Integrity: Merkle tree

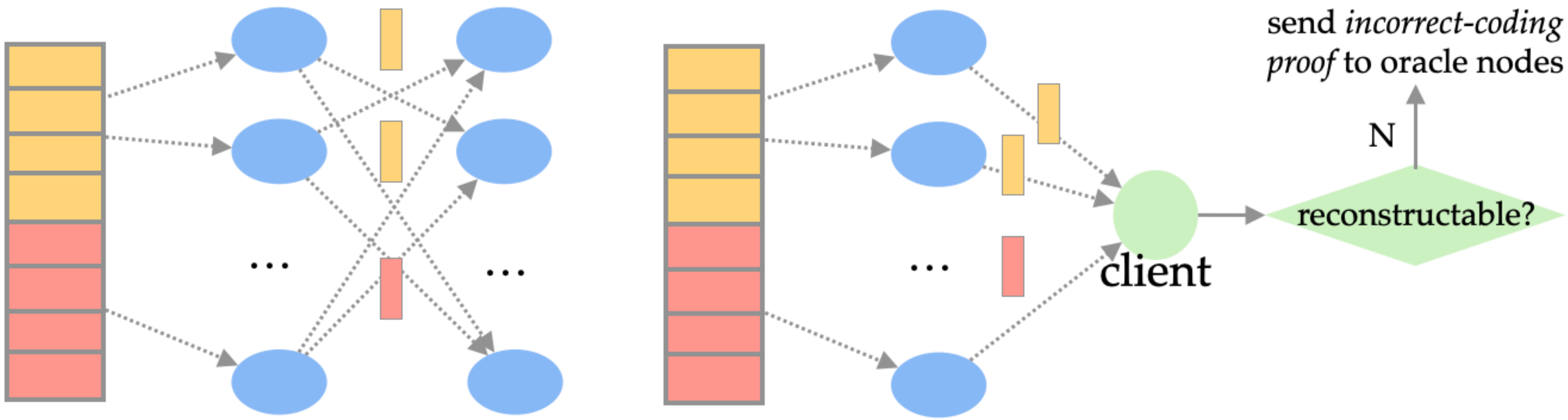
Correctness: incorrect-coding attack



Solutions to Incorrect-coding Attack

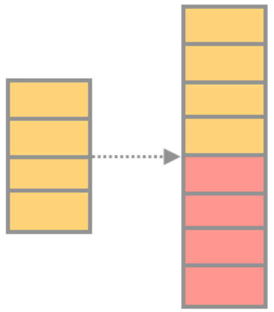




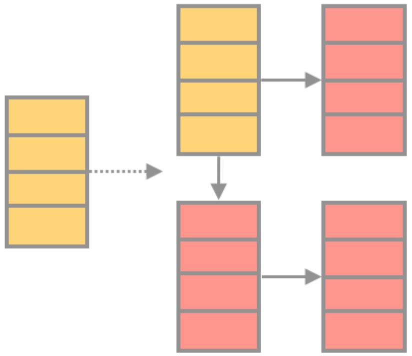


[1] Asynchronous verifiable information dispersal. Cachin, C., Tessaro, S.

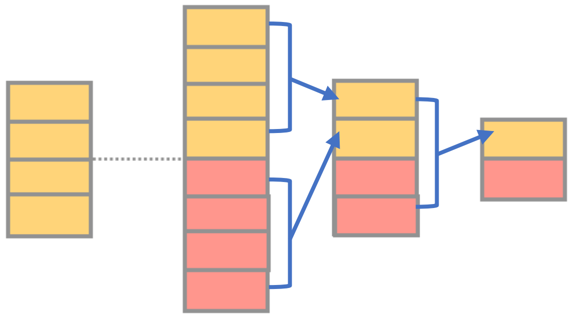
Incorrect-coding Proof



1D-RS



2D-RS[1]

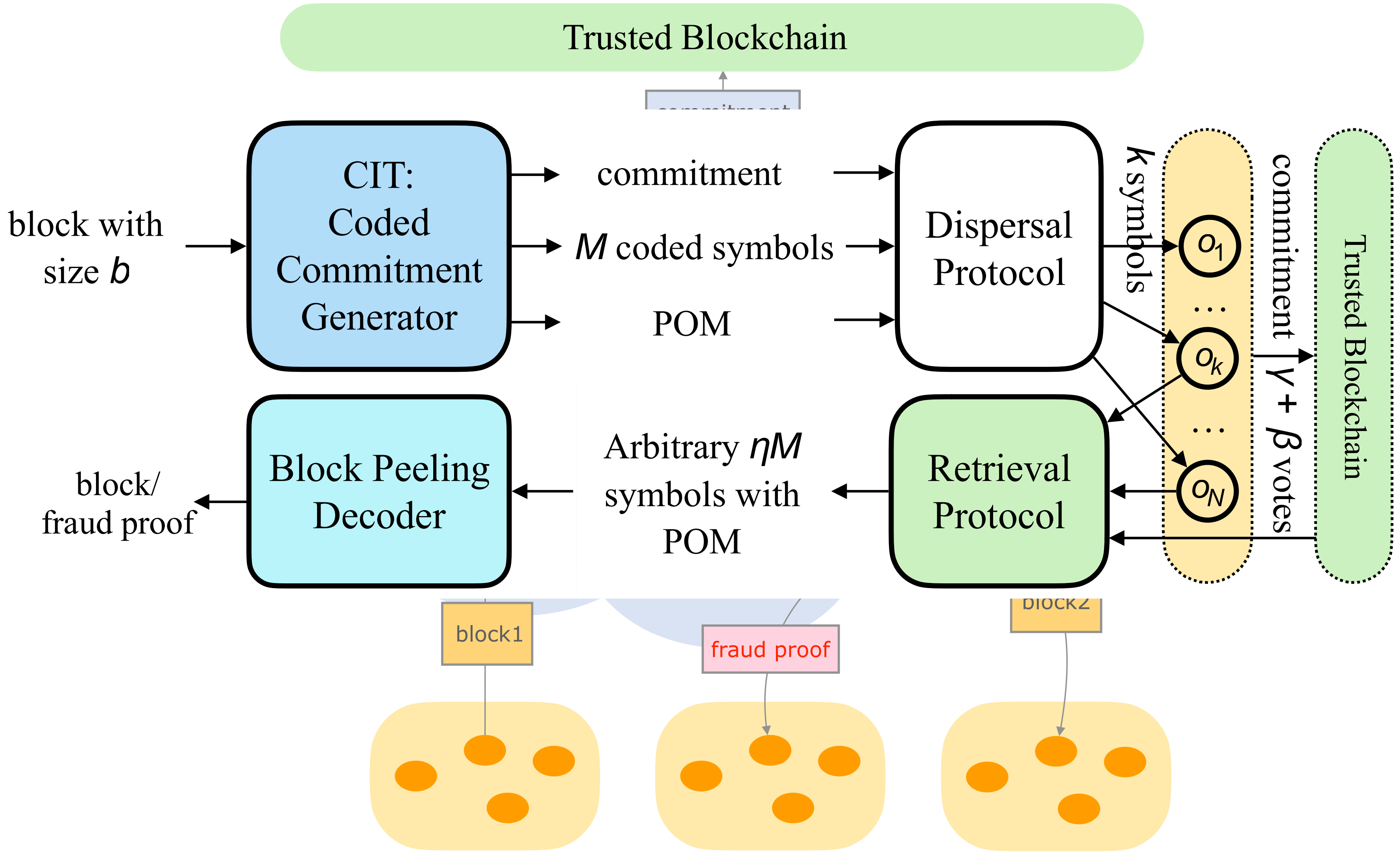


CMT[2]

	1D-RS	2D-RS	CMT
Proof Size	$O(b)$	$O(\sqrt{b}\log b)$	$O(\log b)$

[1] Fraud and data availability proofs: Maximising light client security and scaling blockchains with dishonest majorities. Al-Bassam, M et al
[2] Coded merkle tree: Solving data availability attacks in blockchains. Yu, M et al (FC’20)

Authenticated Coded Dispersal



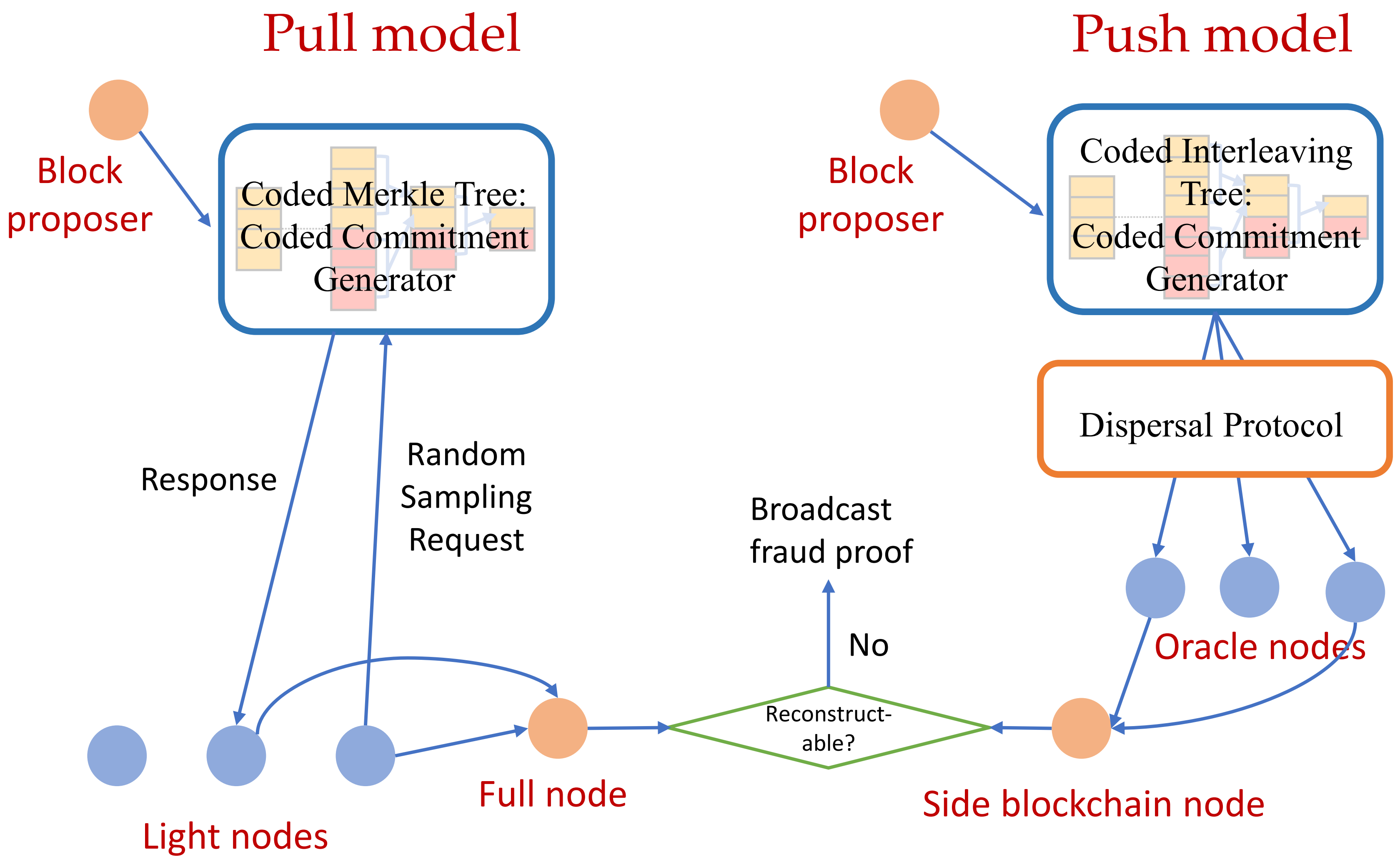
Performance Metrics

Metric	Formula	Range
Maximal adversary fraction	β	[0,1/2]
Storage overhead	$\frac{D_{store}}{D_{info}}$	$[O(1), O(N)]$
Download overhead	$\frac{D_{download}}{D_{data}}$	$[O(1), O(N)]$
Communication complexity	D_{msg}	$[O(b), O(Nb)]$

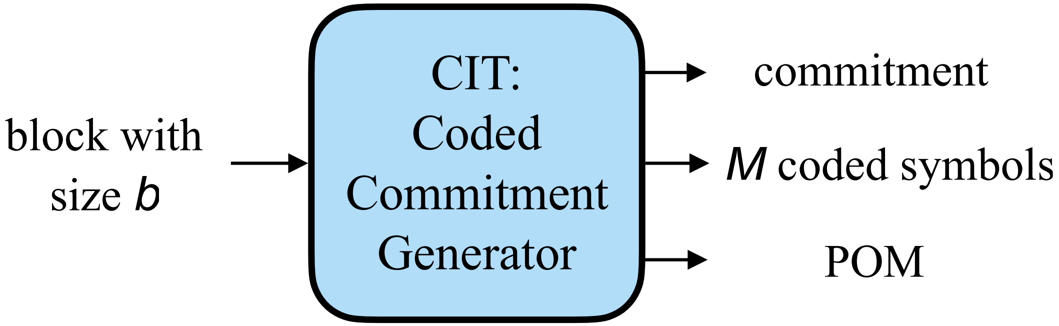
ACeD

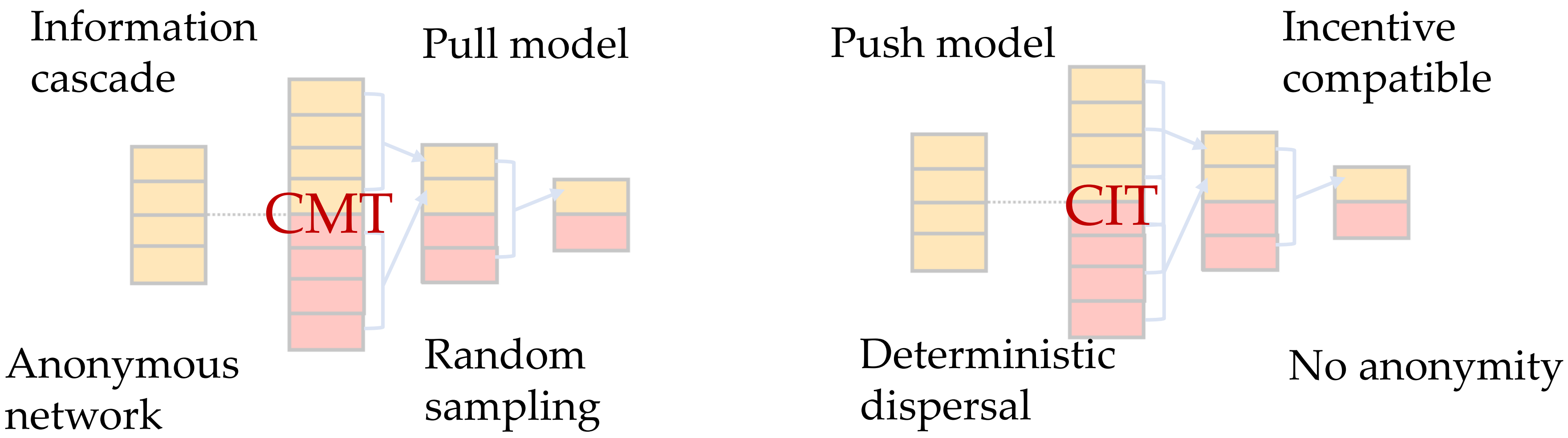
	Maximal adversary fraction	Normal case		Worst case		Communication complexity
		Storage overhead	Download overhead	Storage overhead	Download overhead	
Uncoded (repetition)	$^1/_2$	$O(N)$	$O(1)$	$O(N)$	$O(1)$	$O(Nb)$
Uncoded (dispersal)	$^1/_N$	$O(1)$	$O(1)$	$O(1)$	$O(1)$	$O(b)$
AVID	$^1/_3$	$O(1)$	$O(1)$	$O(1)$	$O(1)$	$O(Nb)$
1D-RS	$^1/_2$	$O(1)$	$O(1)$	$O(b)$	$O(b)$	$O(b)$
2D-RS	$^1/_2$	$O(1)$	$O(1)$	$O(\sqrt{b}\log b)$	$O(\sqrt{b}\log b)$	$O(b)$
ACeD	$\boldsymbol{^1/_2}$	$\boldsymbol{O(1)}$	$\boldsymbol{O(1)}$	$\boldsymbol{O(\log b)}$	$\boldsymbol{O(\log b)}$	$\boldsymbol{O(b)}$

From CMT to CIT

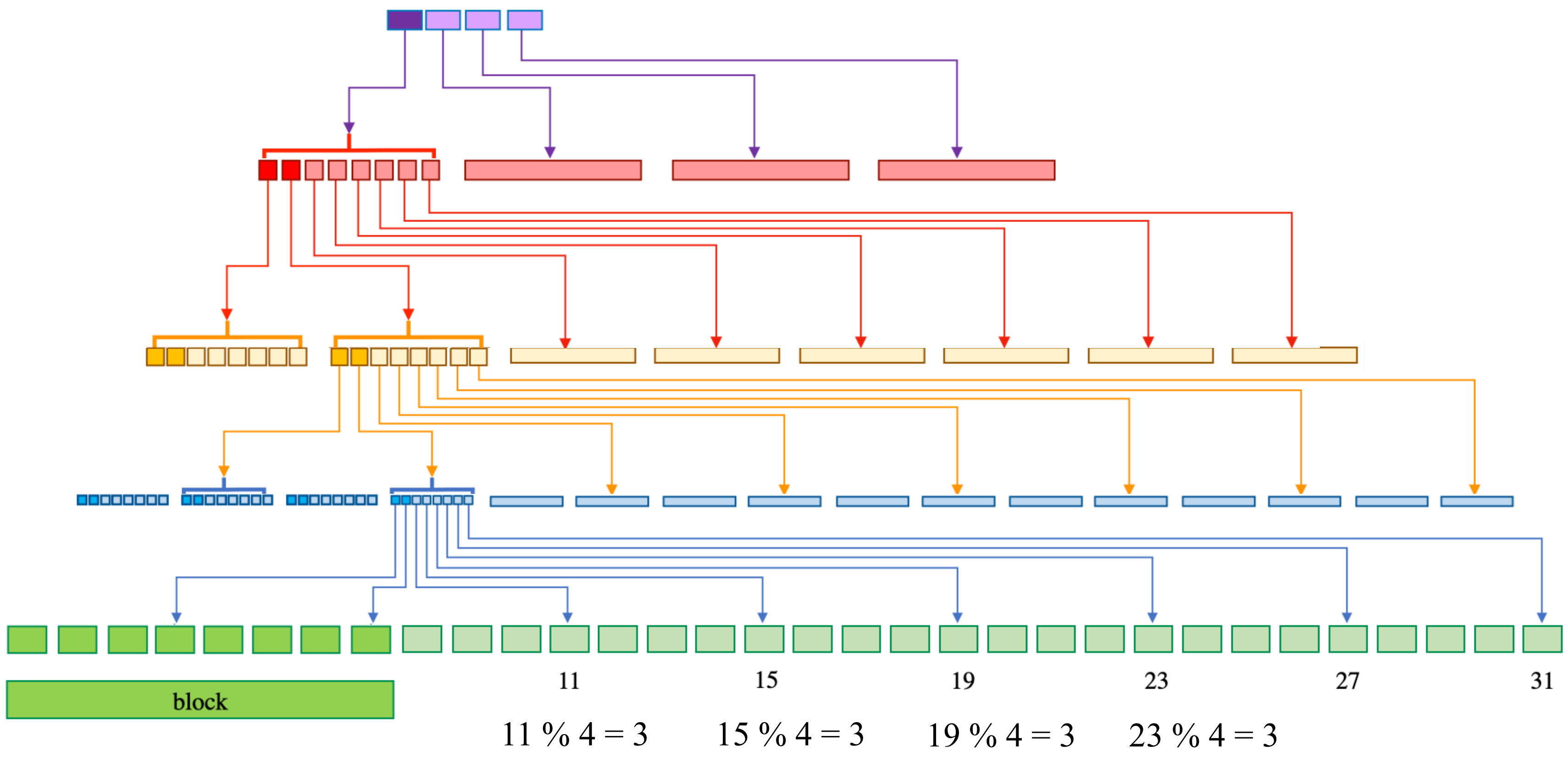


From CMT to CIT



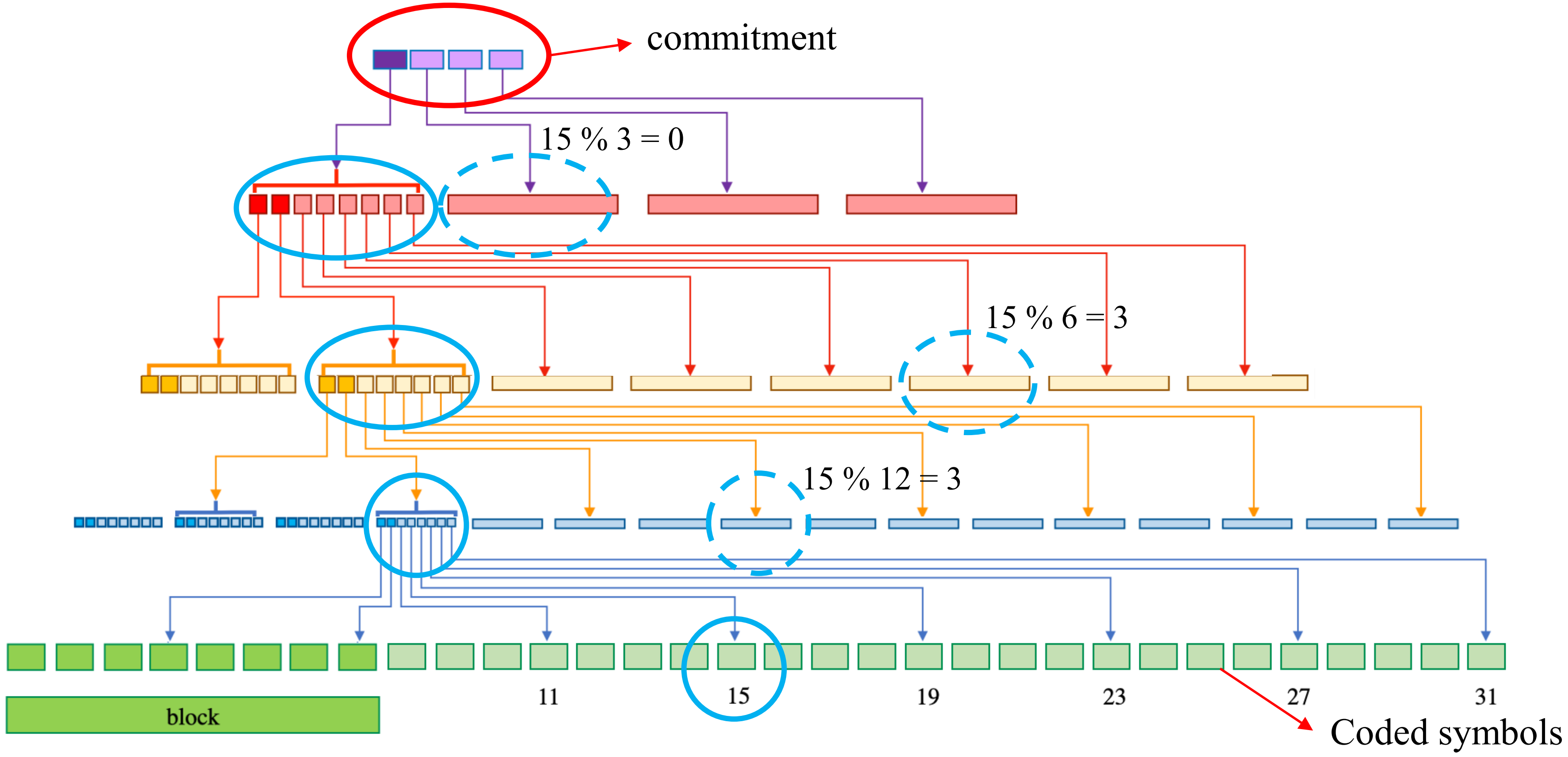


Coded Interleaving Tree



Aggregating rule: parent id $= i$ mod $\#data\ symbols\ in\ parent\ layer$

Coded Interleaving Tree



POM $p(i)$ $e(i)$

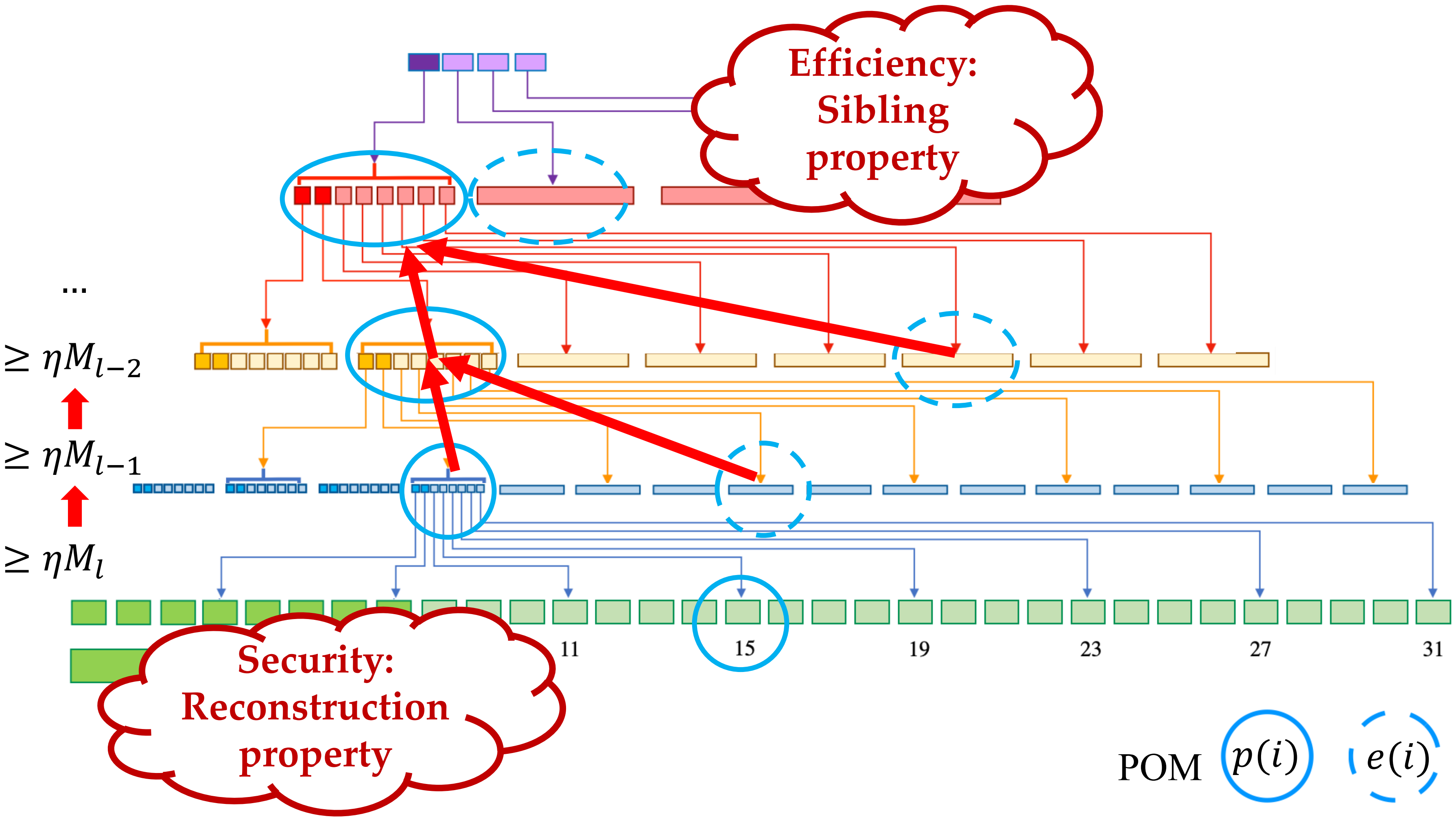
$p(i)$-th data symbol (base 0)

$e(i)$-th parity symbol (base 0)

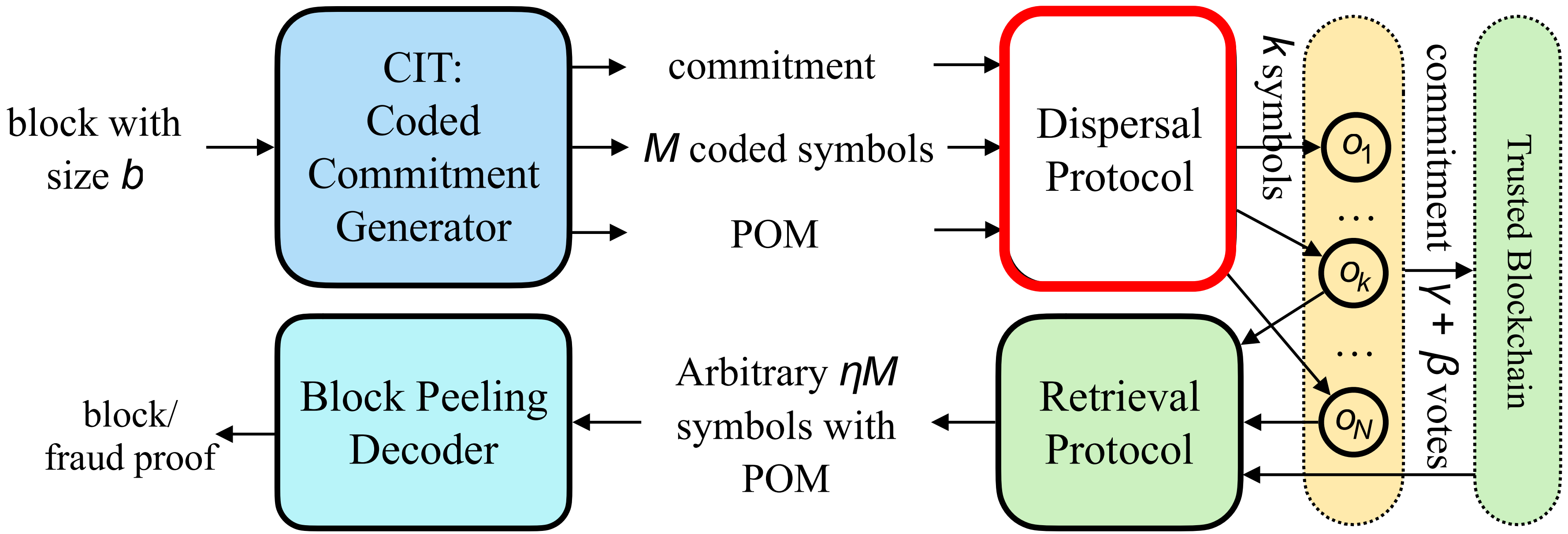
$p(i) = i \bmod \#data\ symbols$

$e(i) = i \bmod \#parity\ symbols$

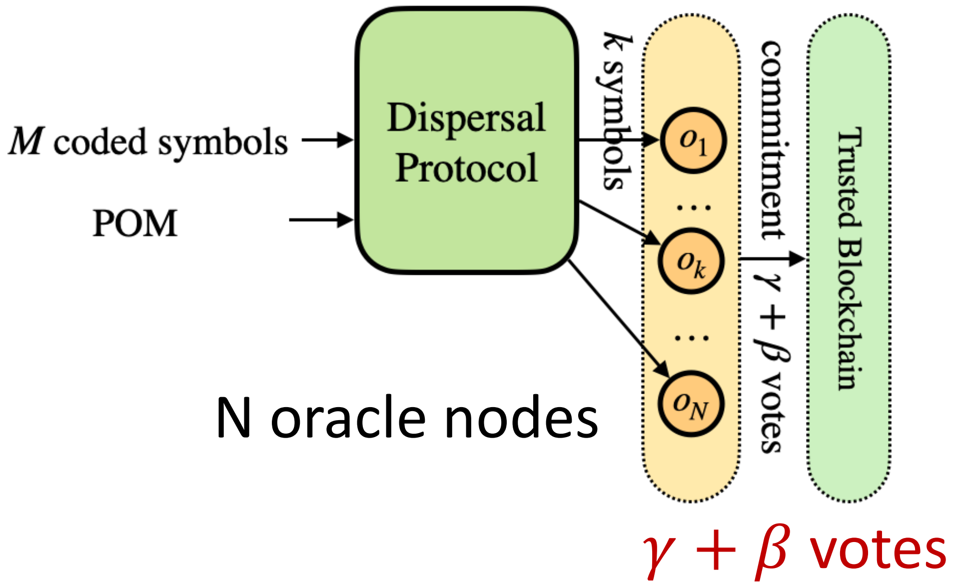
Properties of CIT



Protocol Overview



Dispersal Protocol



Each node:

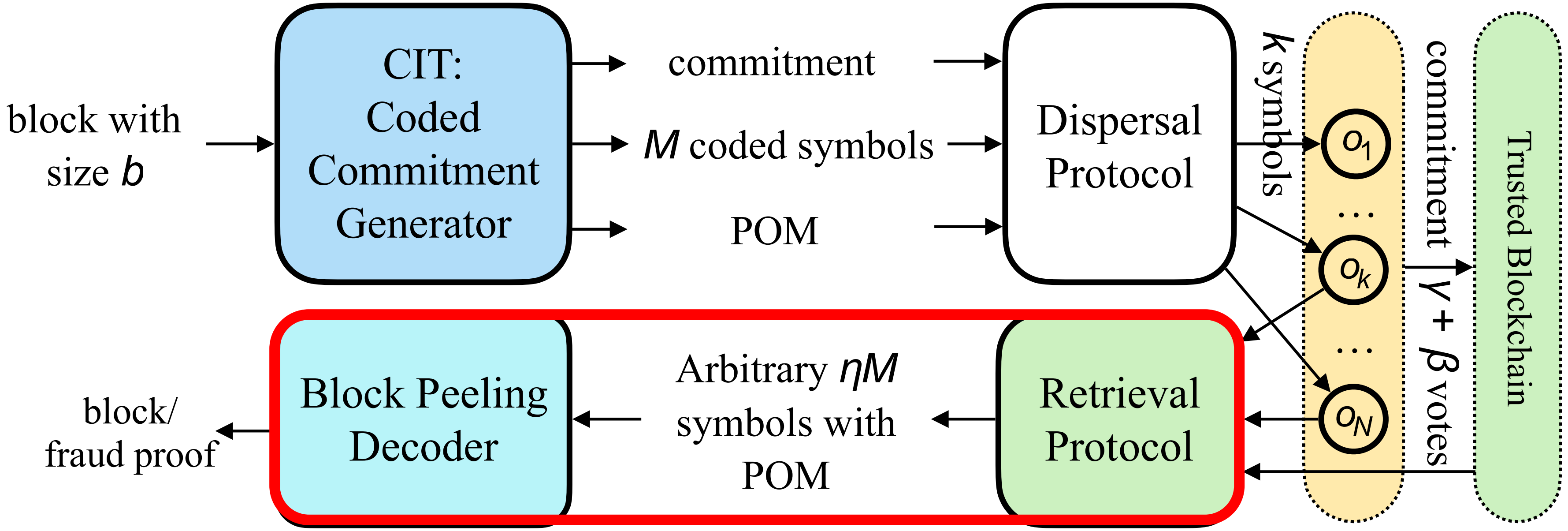
$k = \frac{M}{N\lambda}$ coded symbols

$\geq \gamma N$ oracle nodes:

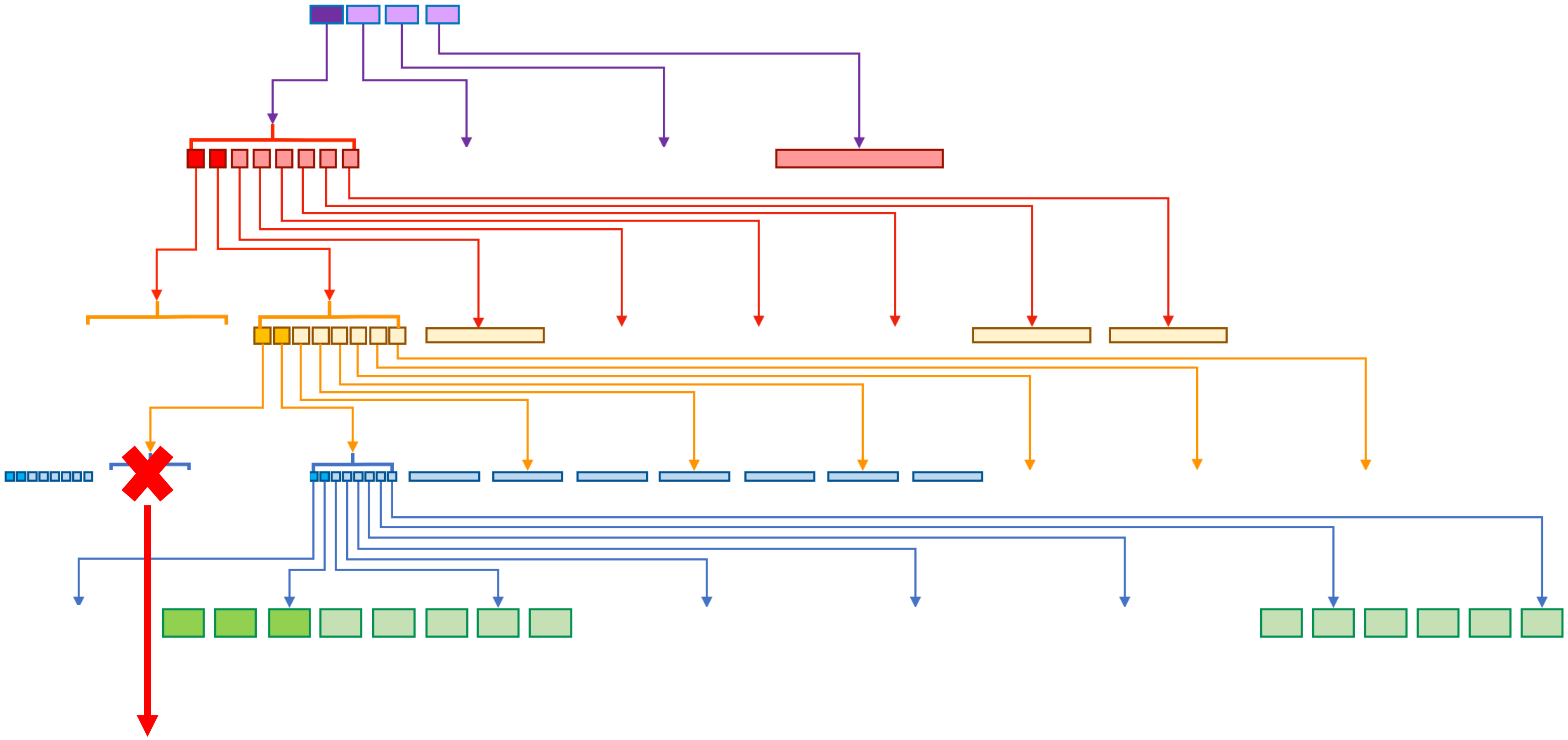
$\geq \eta M$ coded symbols

$\eta > 1 - \alpha$ (undecodable ratio)

Protocol Overview

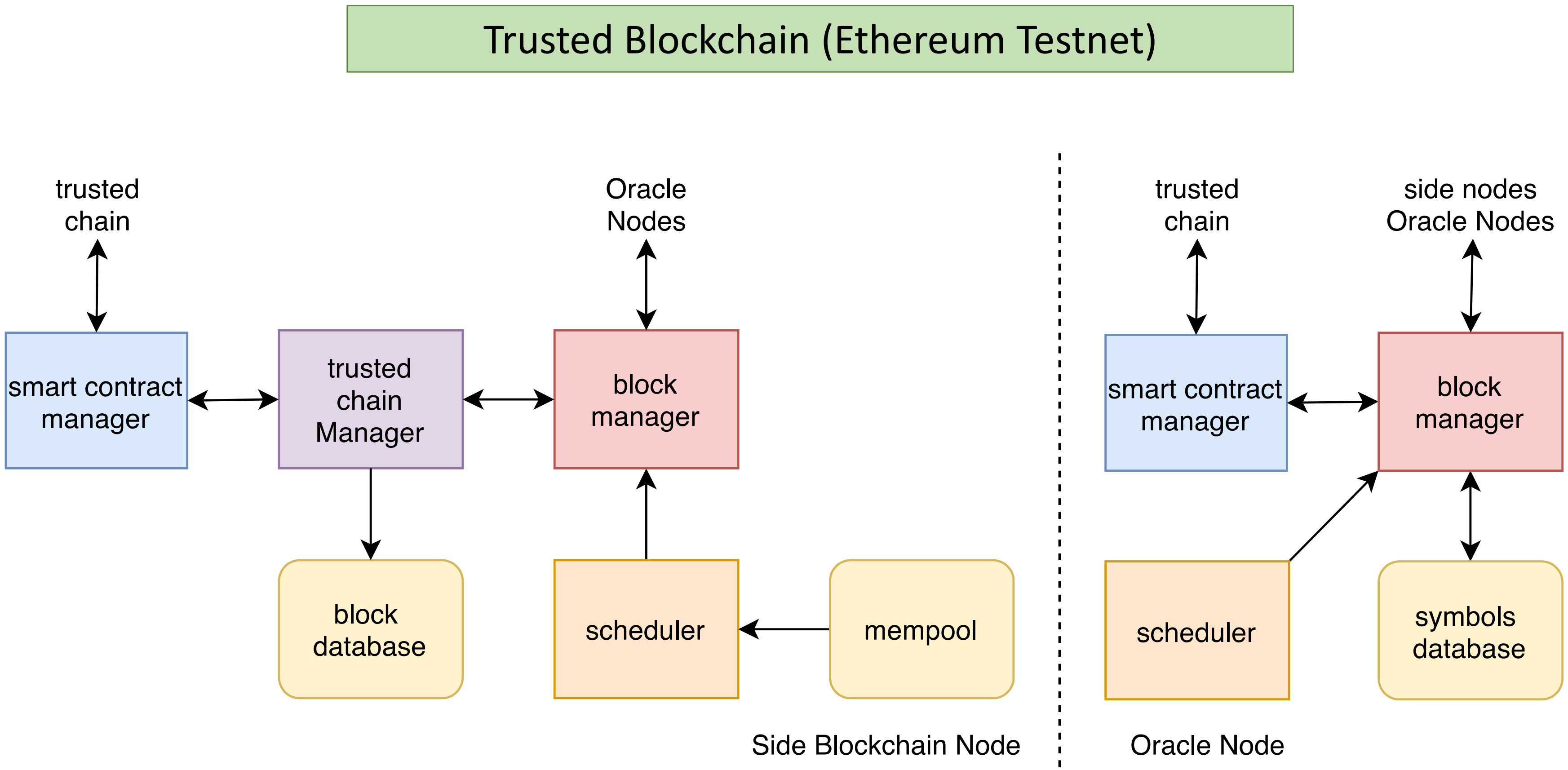


Block Peeling Decoder



Incorrect-coding proof → trusted blockchain

System Modules



https://github.com/simplespy/ACeD

Conclusion

Layer 2 scaling

Sidechains: a simple solution, along with a data availability layer

Original scaling solution of Ethereum

-- Plasma

-- original design of Polygon

-- preferred design of Celestia

Layer 2 Platforms

Payment channels (for Bitcoin)

Rollups (for more general state channels, including Ethereum)

Focus of next two lectures