Blockchain Scaling

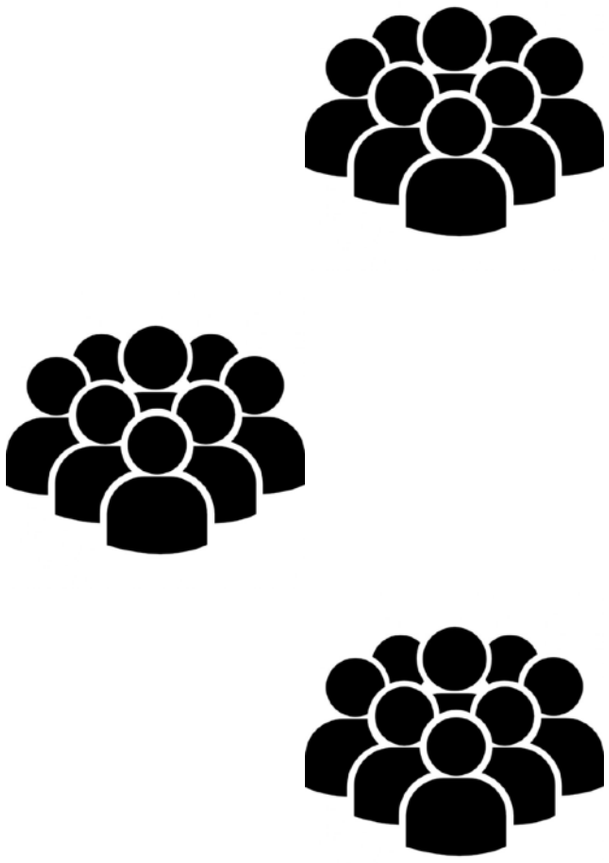
Existing cryptocurrencies do not scale (in terms of TPS)

~ 10 tps

Size of transaction in KB

Complexity of transaction in “Gas”

Why doesn’t it scale?



Many users

Many validators

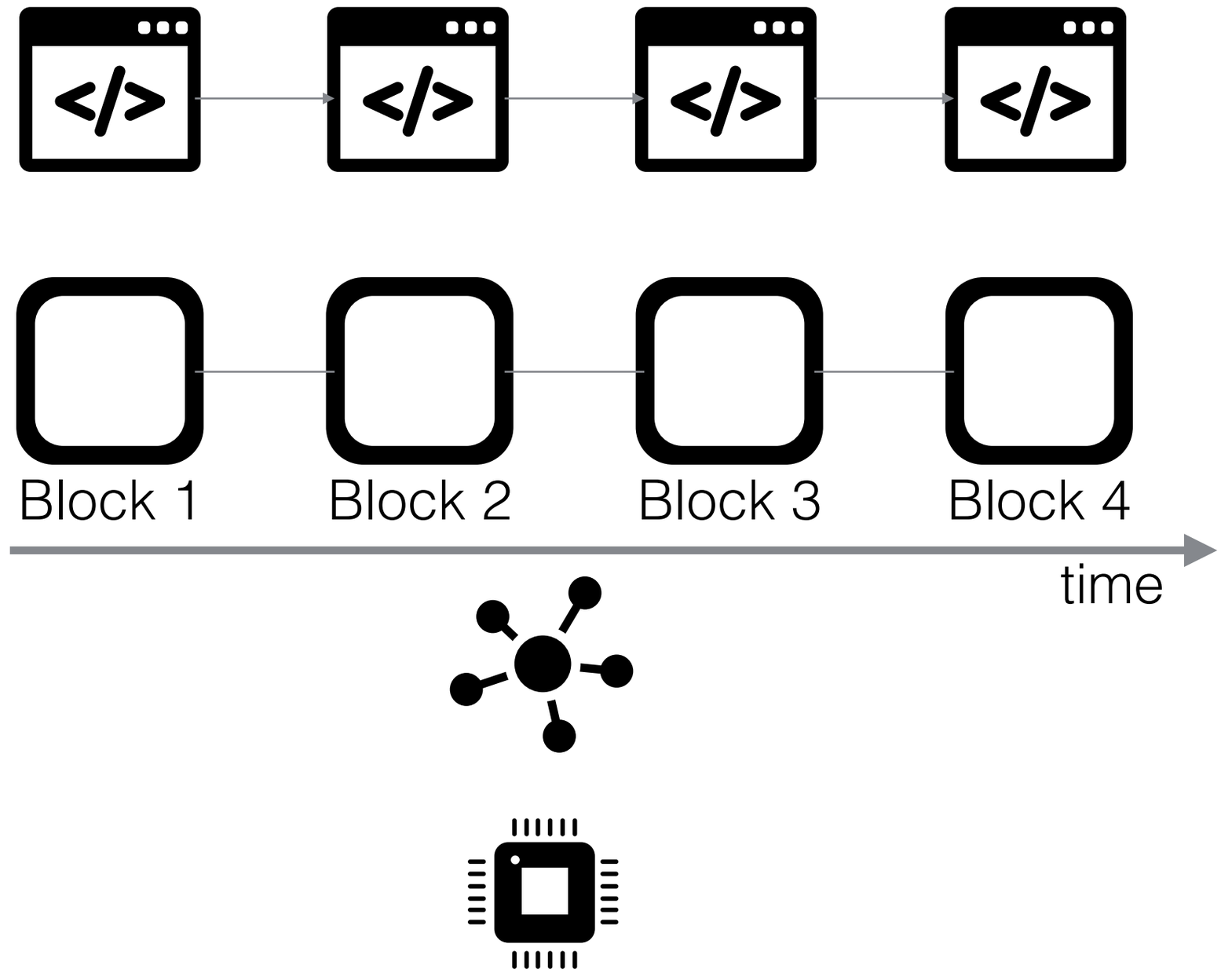
Blockchain Layers

Application
Layer 2

Blockchain
Layer 1

Network
Layer 0

Hardware
Layer -1



How can we scale?

Scaling

- **Network** →  **Comm. latency/bandwidth**
- **Consensus** →  **Tweak Consensus**
- **Sharding** →  **Partitioning the network**
- **Off-Chain** →  **Delayed on-chain settlement**

How can we scale?

Scaling

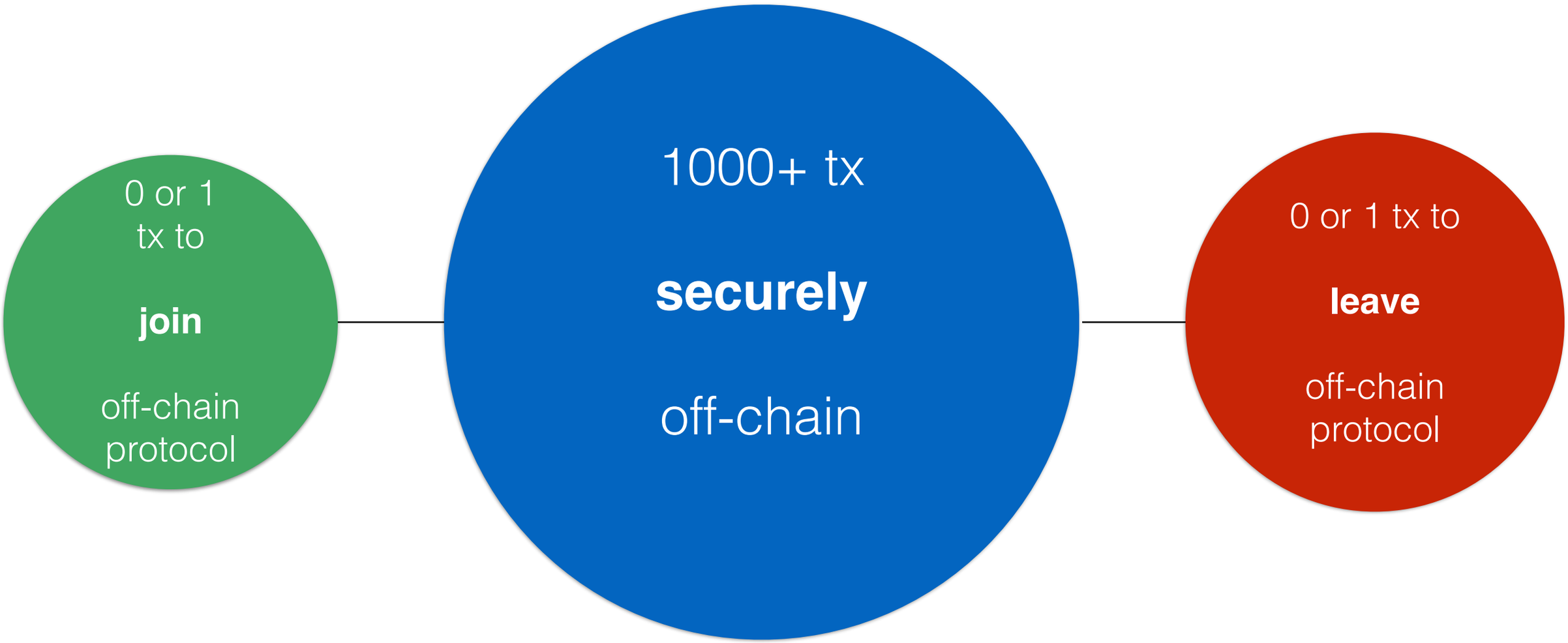
- **Network** → Comm. latency/bandwidth
- **Consensus** → Tweak Consensus
- **Sharding** → Partitioning the network
- **Off-Chain** → **Delayed on-chain settlement**

Complementary!

Off Chain Transaction ==

Transaction outside the blockchain, secured *by* the blockchain

Why is Off-Chain Exciting?



No consensus latency or mining fees,
while still achieving non-custodial security

Backward compatibility!

SoK: Off The Chain Transactions

Lewis Gudgeon
Imperial College London
l.gudgeon18@imperial.ac.uk

Pedro Moreno-Sanchez
TU Wien
pedro.sanchez@tuwien.ac.at

Stefanie Roos
TU Delft
s.roos@tudelft.nl

Patrick McCorry
King's College London
patrick.mccorry@kcl.ac.uk

Arthur Gervais
Imperial College London, Liquidity Network,
Lucerne University of Applied Sciences and Arts
a.gervais@imperial.ac.uk

***Abstract*—Blockchains have the potential to revolutionize markets and services, yet, currently exhibit high latencies and fail to handle loads comparable to those managed by traditional custodian financial systems. *Layer-two* protocols, built on top of (*layer-one*) blockchains, avoid disseminating every transaction to the whole network by sending transactions *off-chain* and instead utilize the blockchain only as a recourse for disputes. The promise of layer-two protocols is to complete transactions in sub-seconds, reduce fees, and allow blockchains to scale.**

With this Systematization of Knowledge, we are the first to structure the complete rich and multifaceted body of research on layer-two transactions. Categorizing the research into payment and state channels as well as commit-chains, we provide a comparison of the protocols and their properties. We contribute a systematization of the associated synchronization and routing protocols along with their privacy and security aspects. Contrary to common belief in the blockchain community, we show that layer-two can scale blockchains; that layer-two protocols are secure without full collateralization; that privacy of layer-two transaction is not granted by default; and that fees depend on the transmitted transaction value. The SoK clears the layer-two fog, highlights the potential of layer-two solutions and identifies their unsolved challenges and promising avenues of future work.

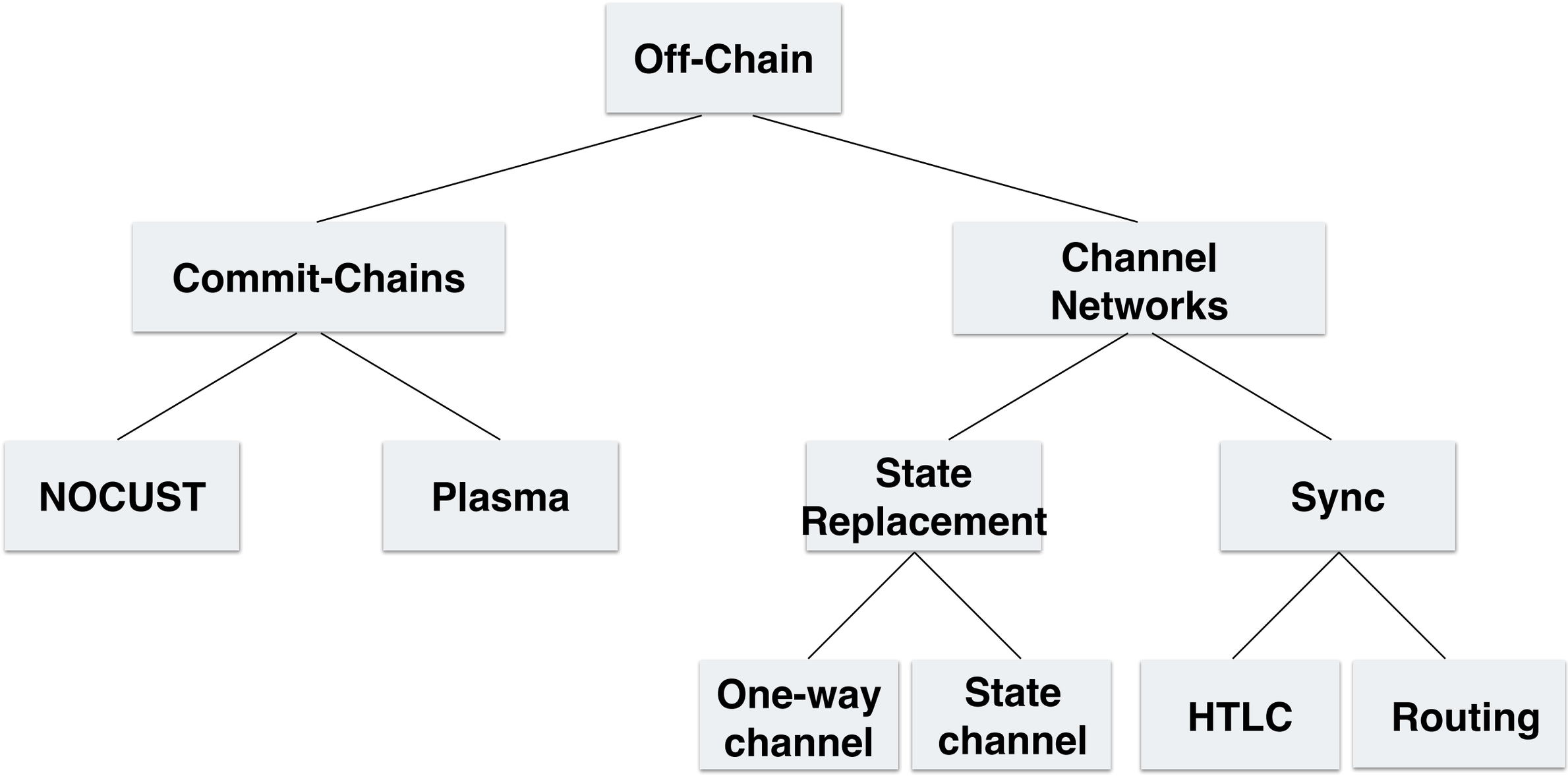
I. Introduction

The advent of blockchains over a decade ago [1]–[4] spurred rapid and extensive innovation across different scientific disciplines. Blockchains offer a mechanism through which mutually mistrusting entities can cooperate in the absence of a trusted third party. However, the use of broadcast in those non-custodial protocols limits their scalability to about ten transactions-per-second (tps) [5], [6], compared to custodian payment systems with thousands of tps [7]. Scaling limitadecentralized structure. Consensus changes might even lead to different, forked systems [23]. Layer-two protocols enable users to perform so-called *off-chain* transactions through private communication, rather than broadcasting the transaction on the (parent) blockchain. This optimization reduces the transaction load on the underlying blockchain and is fully backward compatible. The theoretical transaction throughput is only bounded by the communication bandwidth and latency of the involved parties. Off-chain transaction security can be guaranteed via allocated collateral, e.g. in payment channel designs [24]–[27] or by offering delayed transaction finality in commit-chain proposals [28].

A. This Systematization of Knowledge

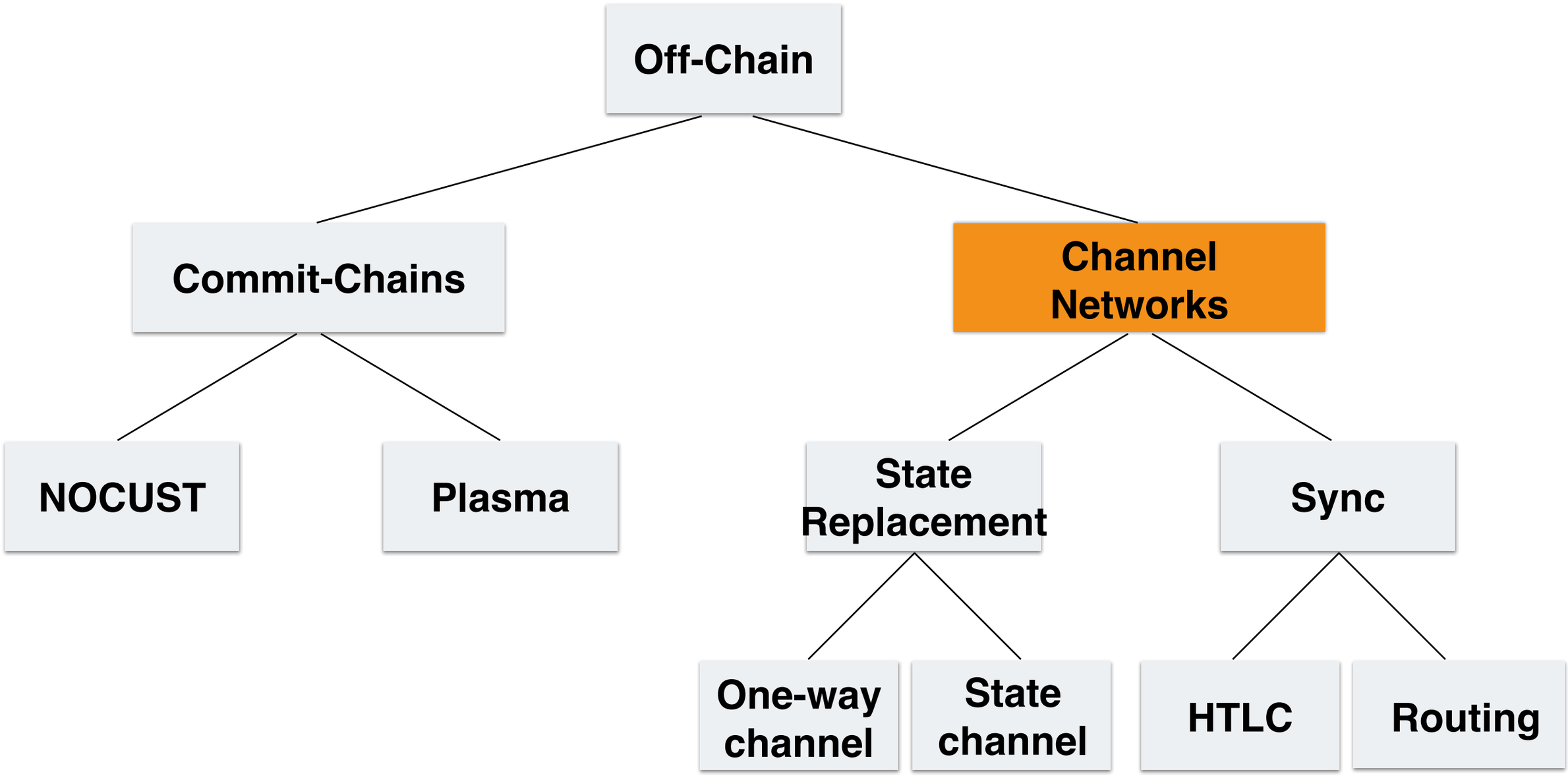
A rich body of literature has emerged on off-chain protocols, proposing payment [24]–[27], [29], state [30] and virtual [31] channels, payment channel networks (PCNs) [27], [29] and related routing protocols [32]–[37], channel rebalancing [38] and channel factories [39] constructions, commit-chains [28], [40], channel hubs [41], [42], privacy-enhancing channels [41], [43]–[45]. However, the sources of information about layer-two protocols are highly disparate. Moreover, in part due to the rapid pace of advancement in the blockchain field, we observe, mostly outside academia, a frequent under-specification of constructions and their adversarial assumptions. This makes it exceedingly difficult to discern thought-through concepts from marketing activities. We aim to clear the fog surrounding layer-two protocols, equipping newcomers to this inaccessible field with a concise reference, and inform the directions of future

Which Off-Chain Solution?

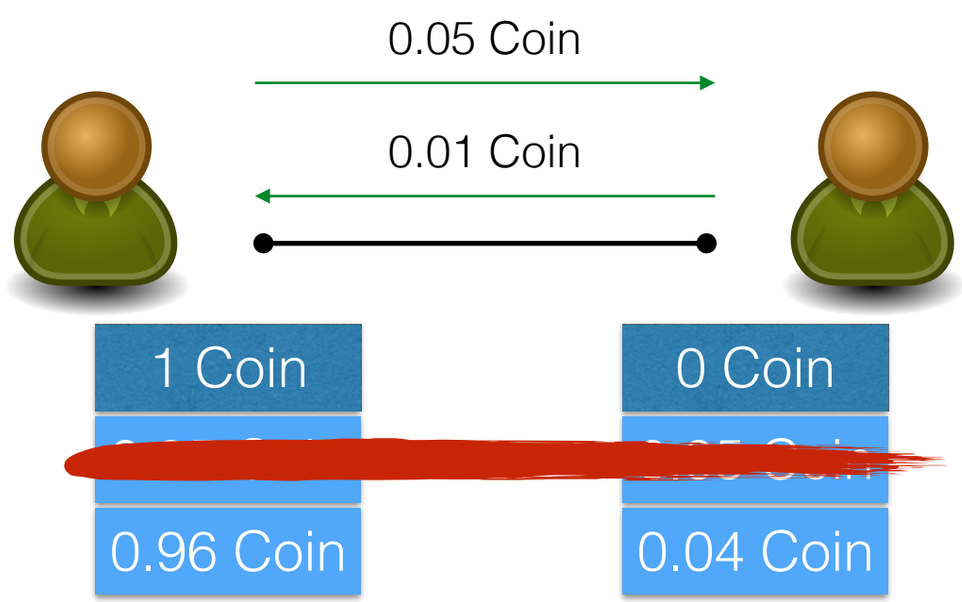


Channel Networks

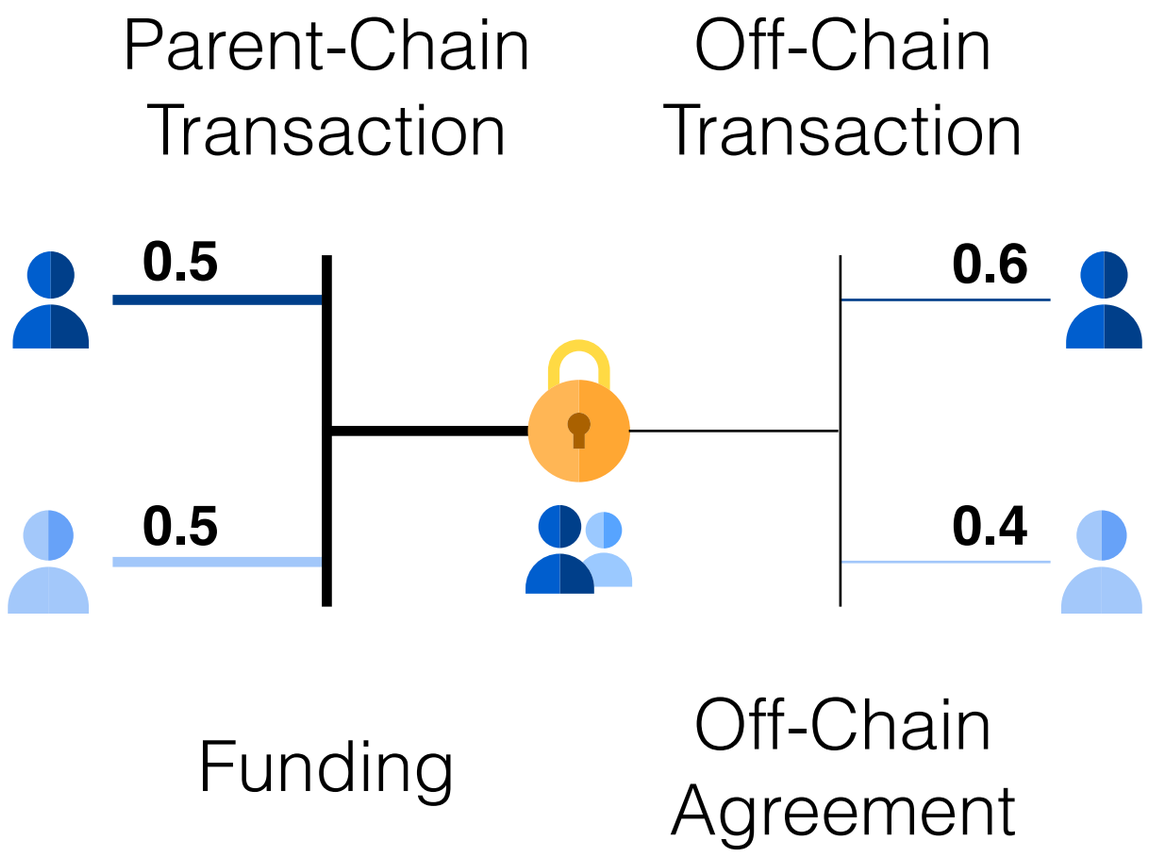
Which Off-Chain Solution?



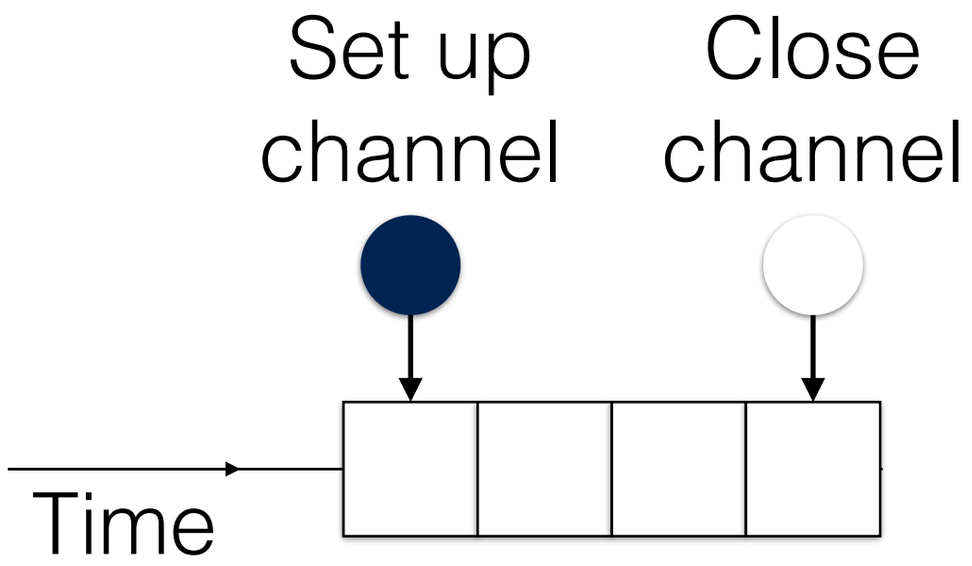
Payment Channel



Payment Channel



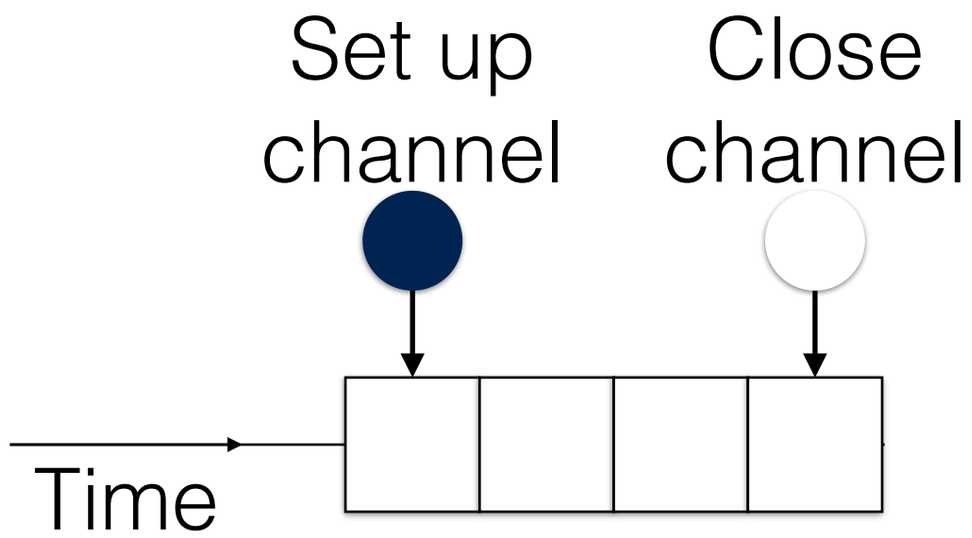
Payment Channel Properties



- **Balance Security**:
  Any party can always withdraw the agreed balance on-chain with a dispute.

- **State Progression**:
  Any party can enforce a state transition on-chain to reach a terminal state.

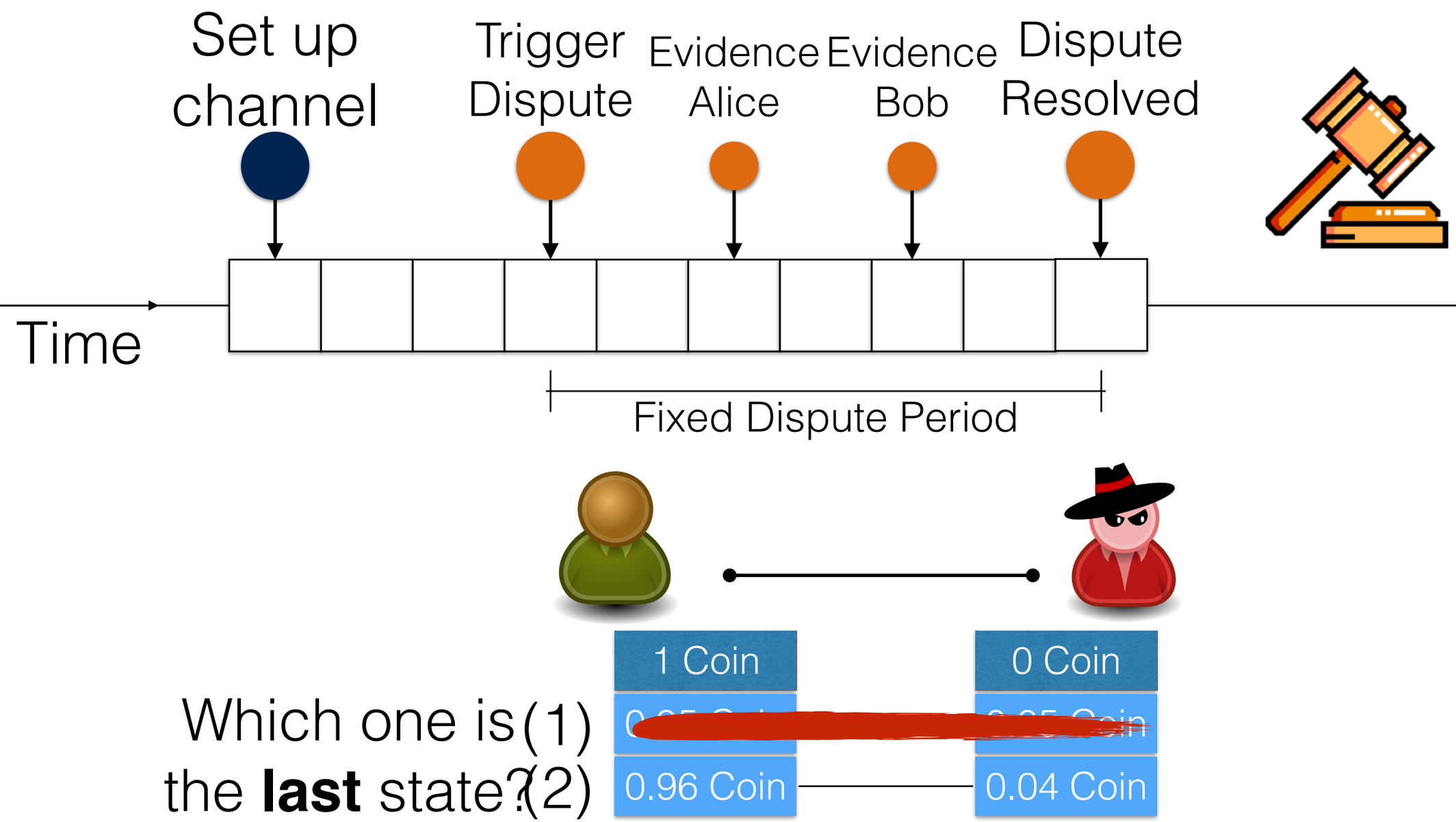
Payment Channel



- Unanimous Establishment
- Unanimous Transition (without dispute)

What if the parties do not agree?

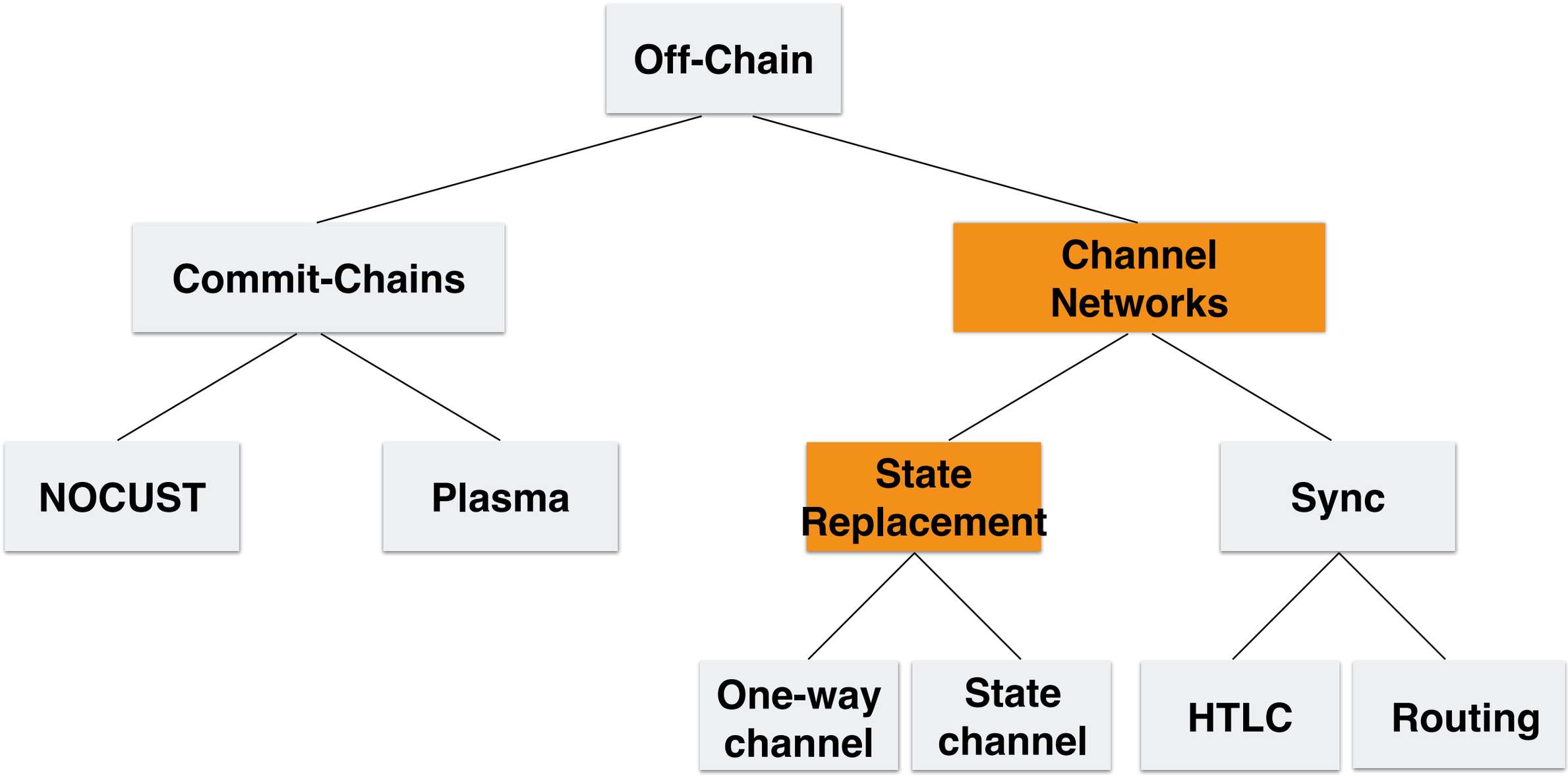
Payment Channel - Dispute



Which one is (1) the **last** state? (2)

Channel Networks State Replacement

Which Off-Chain Solution?



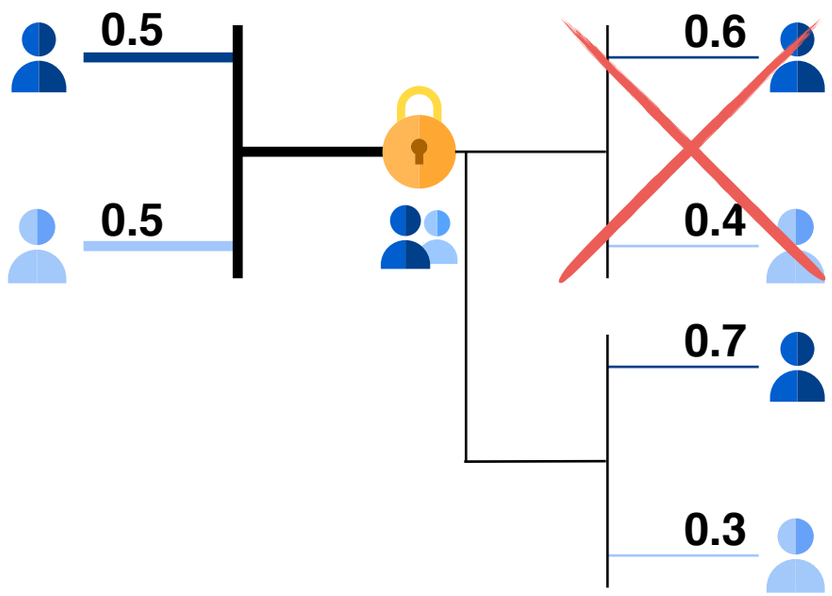
Payment Channel (redistribution of assets)

Spilman	Duplex Micropayment	Lightning

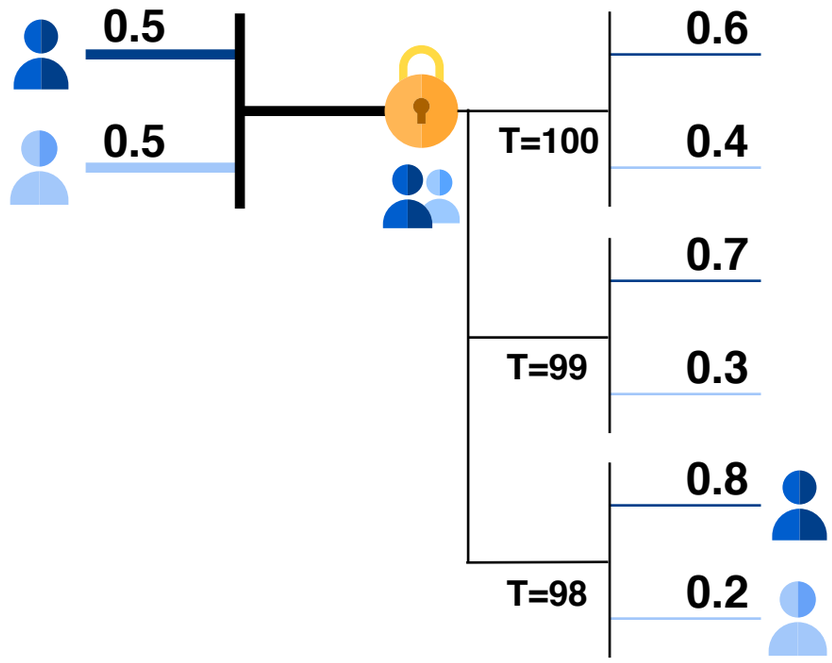
State Channel (game, voting, auctions, etc)

Sprites	Perun	Kitsune

State Replacement

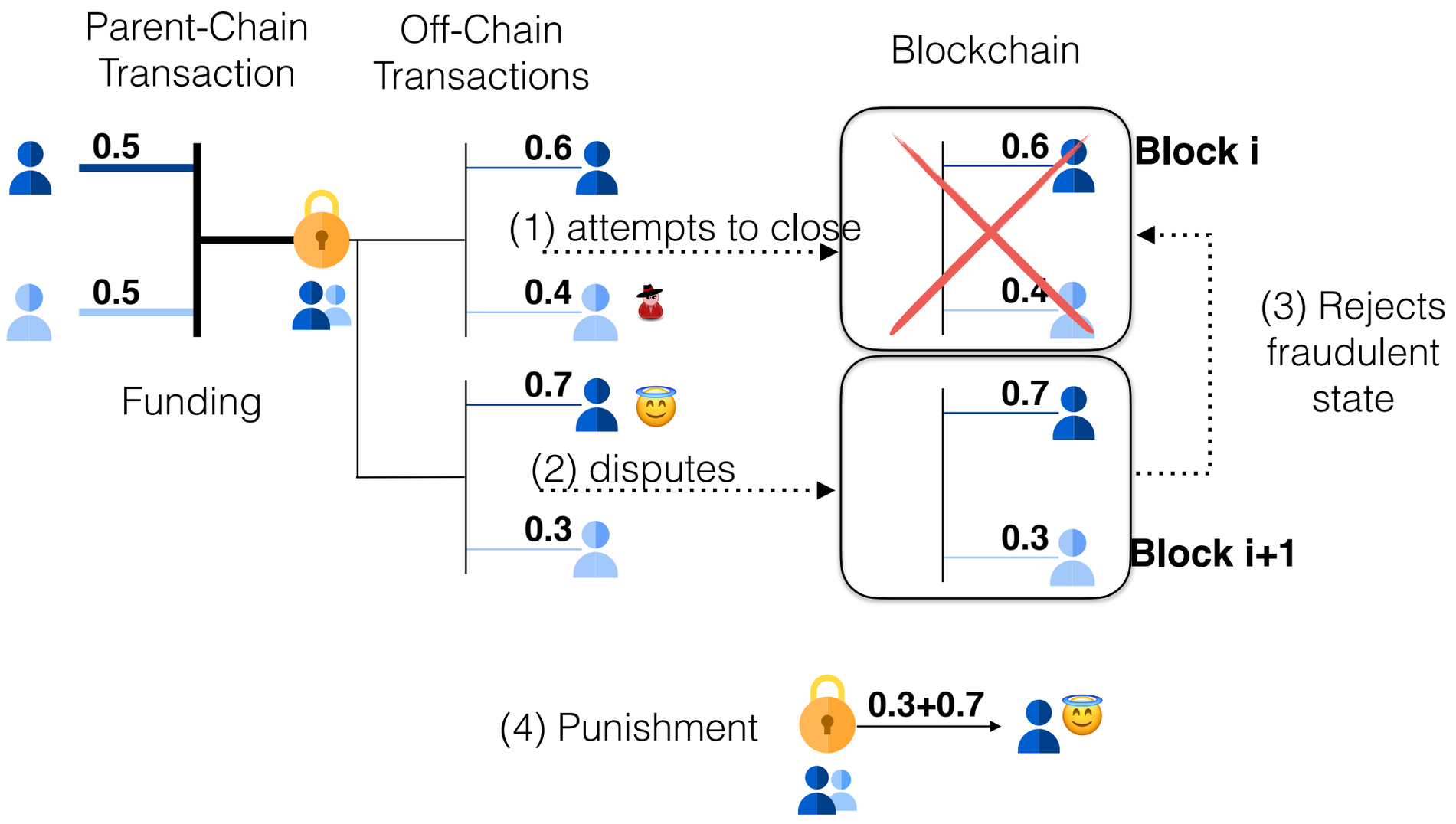


Time Lock State Replacement



Lowest timelock is first included in the blockchain.

Revocation State Replacement (Lightning)



State Replacement Techniques (2013 - 2015)

Replace by..

Incentive → **One-way payments**

Spilman

Receiver signs and publishes final statement

Time Lock → **Bi-directional**

Duplex Micropayment Channels

Tension between throughput and on-chain costs
Expiry time (later removed)

Revocation → **Bi-directional, no expiry**

Lightning

Agree on last state, keep all previous revoked states

Protocol Complexity

Bitcoin's model makes it challenging to remove expiry time and throughput limitations

State Replacement Techniques (2016+)

Replace by..

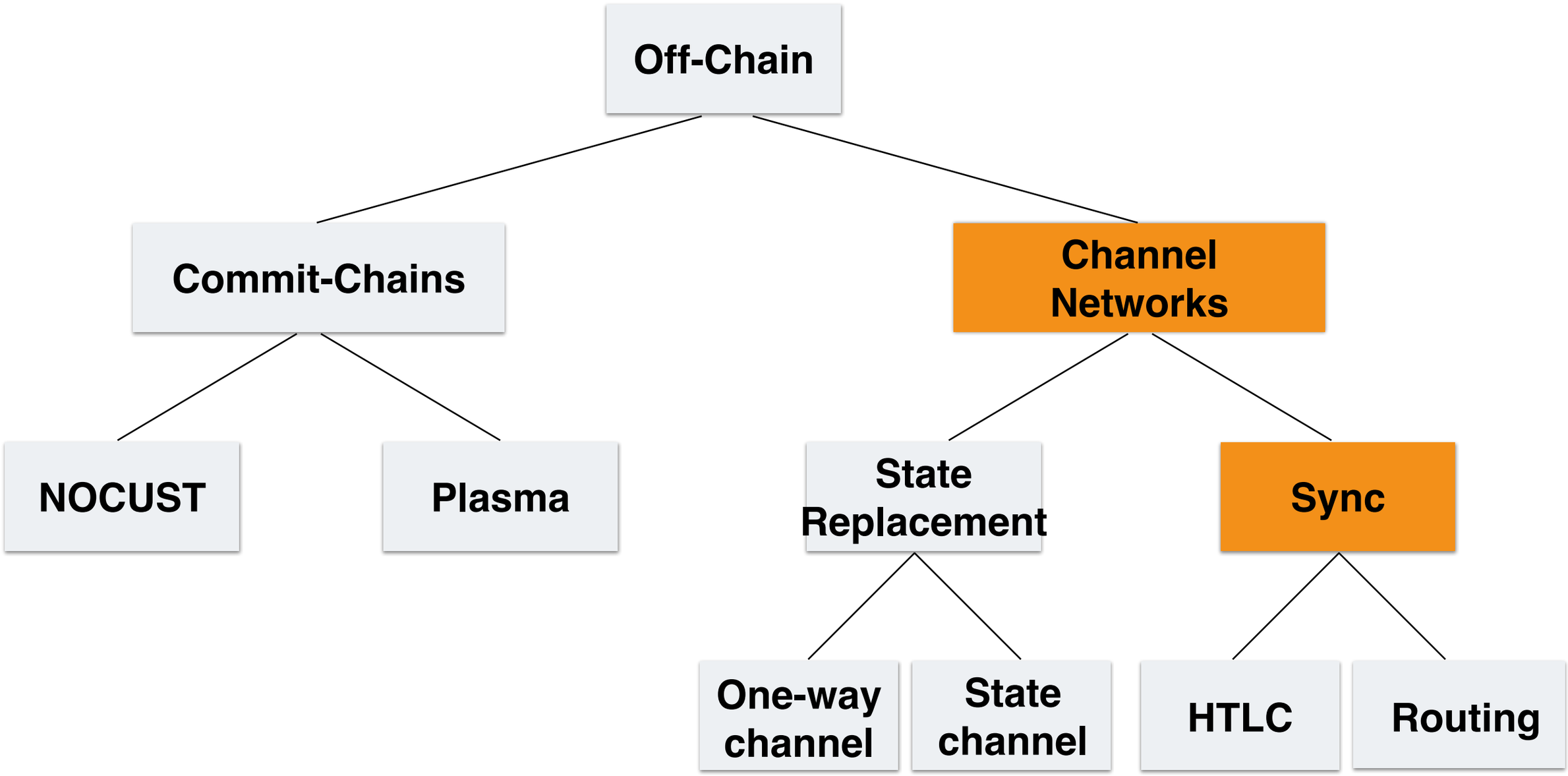
Version ⟶ **State change, everyone signs**

Increment version

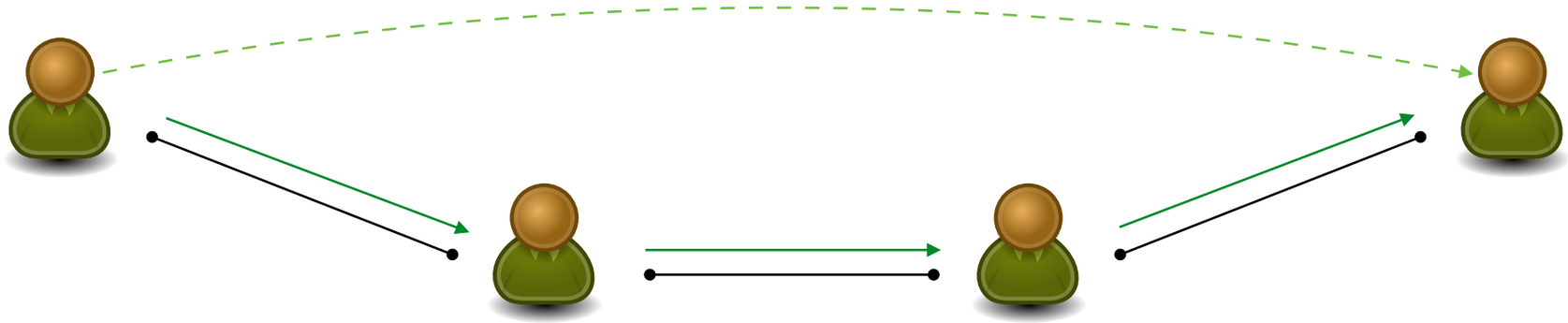
Sig(A), Sig(B), Hash of State(i), i

Channel Networks
HTLC

Which Off-Chain Solution?

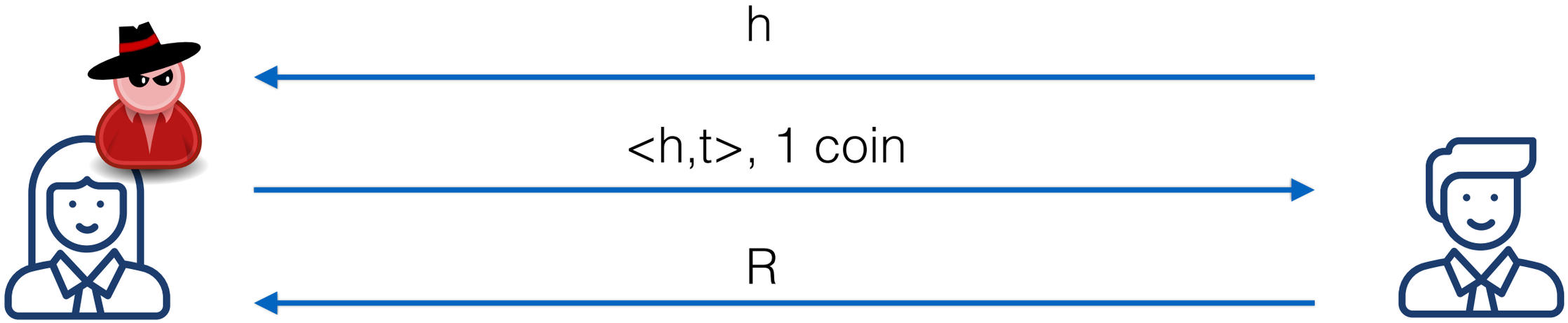


Payment Channel Network



We need a **simple conditional transfer**.

A Humble Conditional Transfer



(1) Jackson computes H(R) and shares h

(2) Michaela's conditional transfer: *"Jackson, if you reveal R before time t, you get the coins"*

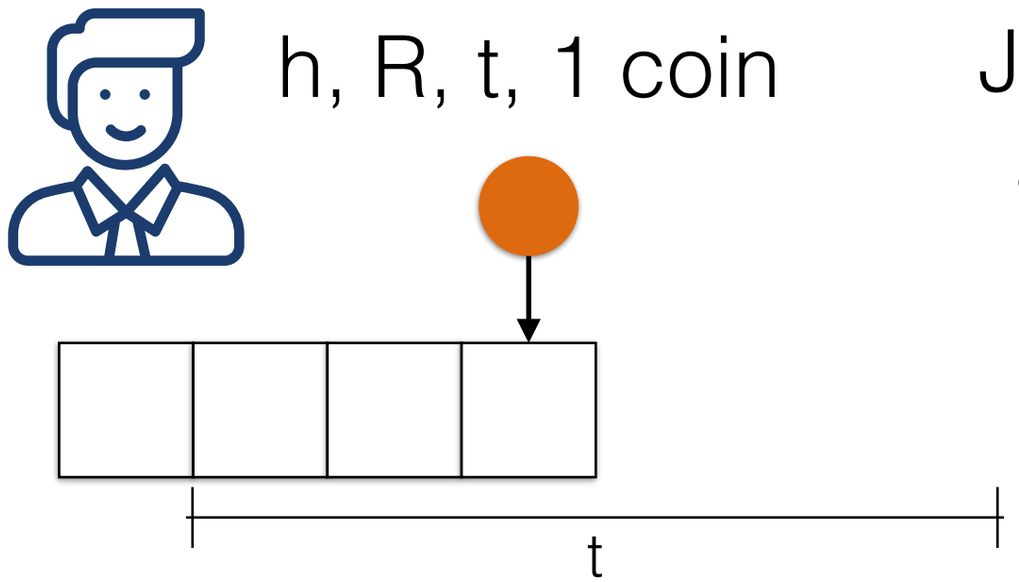
(3) Jackson reveals R

(4) Michaela cooperates

..what if Michaela doesn't cooperate?

Hashed Time-locked Contract

== conditional transfer + evidence

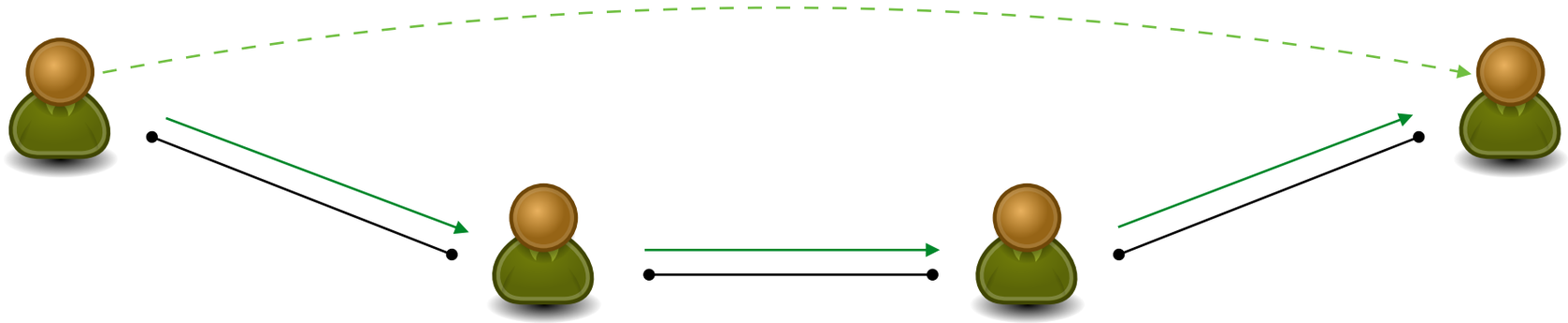


Evidence
h, R, t, 1 coin

The blockchain confirms that Jackson revealed R before time t and complied with the contract

-> He receives the coin.

Payment Channel Networks



Path-based Payments (HTLC)

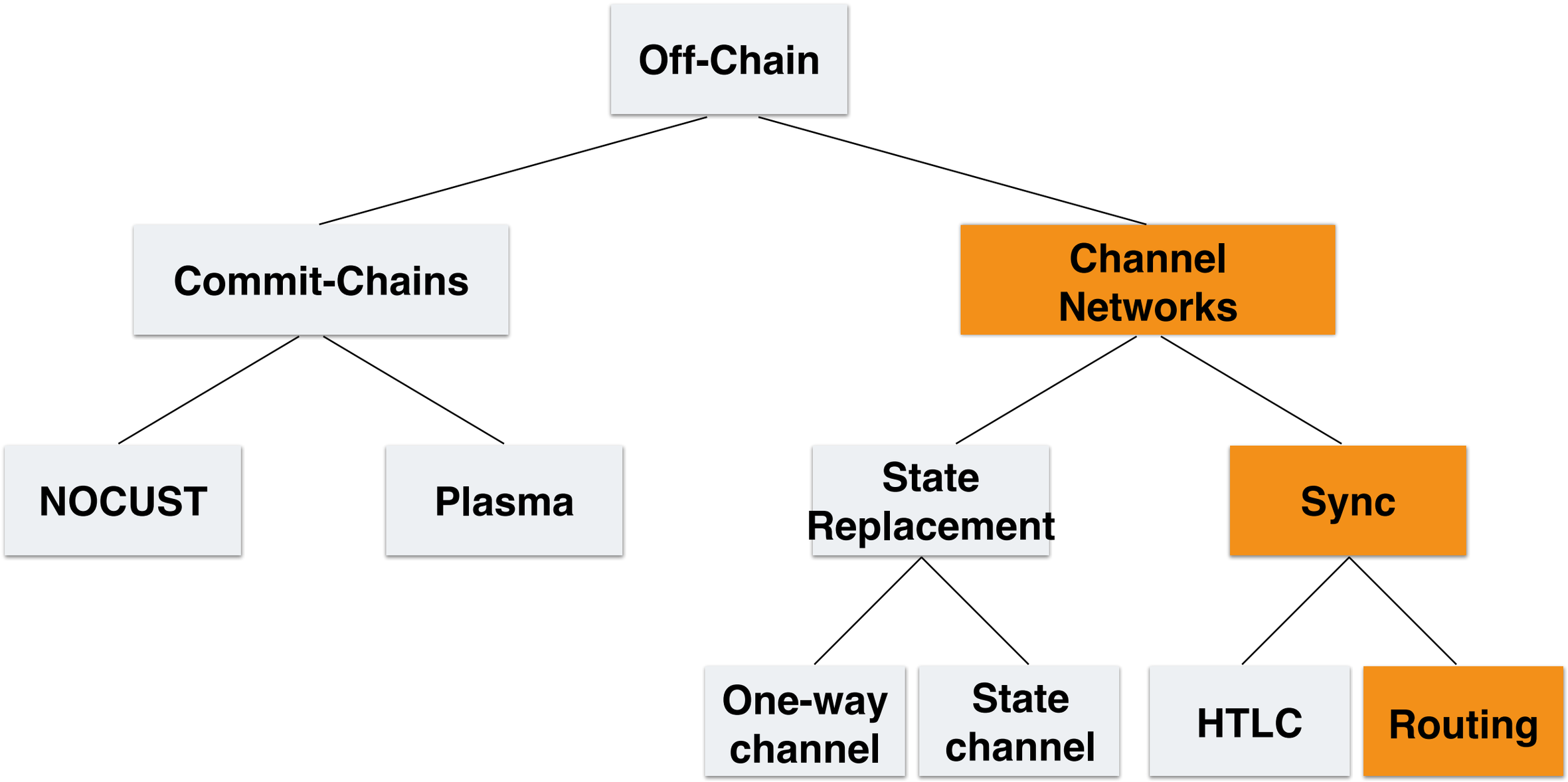
We can synchronise a payment among peers on the path!

Path-based virtual Payment Channels (Perun)

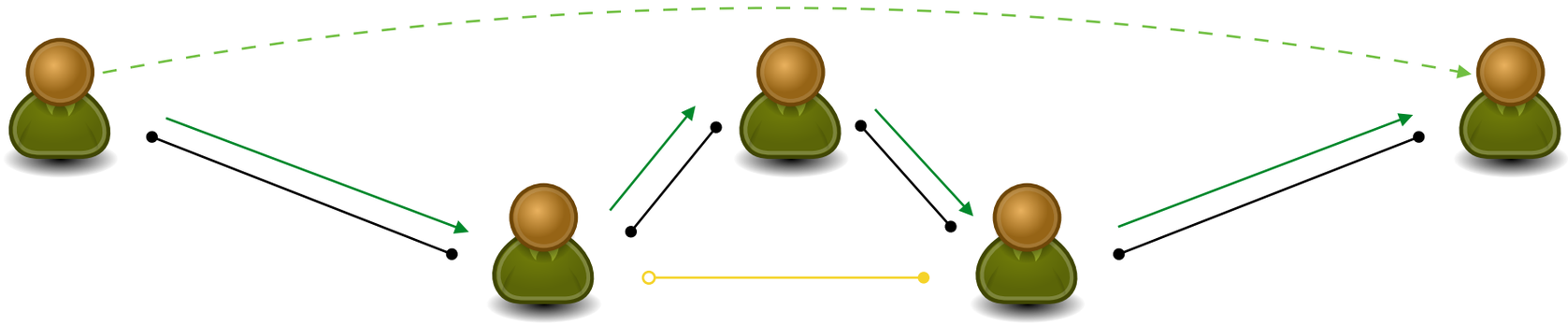
We can set up a virtual channel to avoid intermediary peers to be responsive!

Channel Networks
Routing

Which Off-Chain Solution?

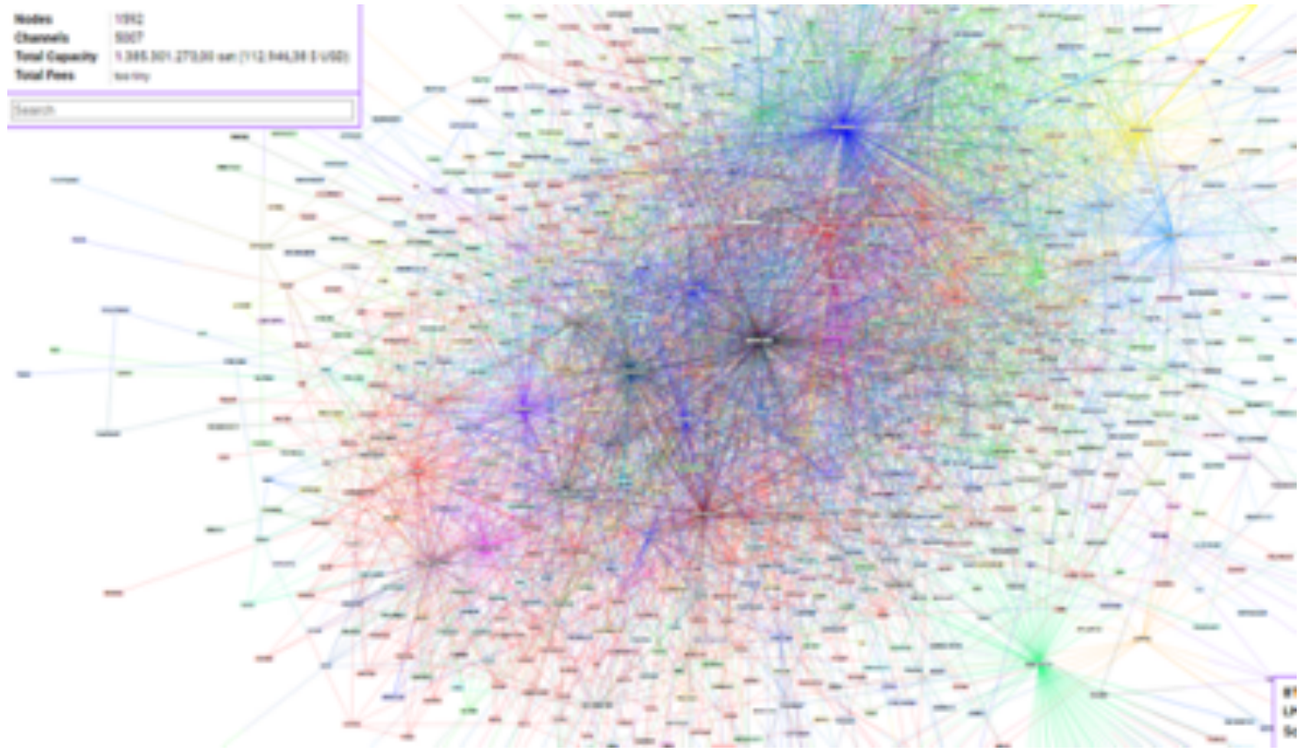


Payment Channel Network - Routing



Route Finding

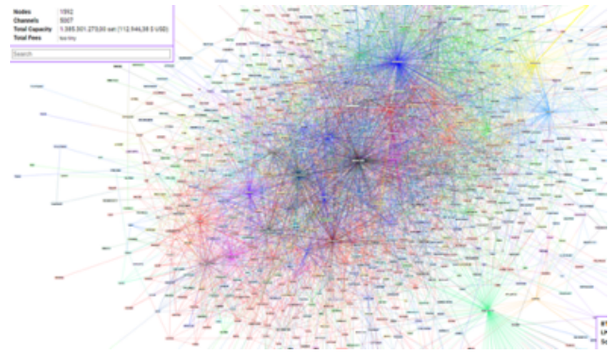
Payment Channel Network - Routing



The blockchain keeps a **record** of which channels are open

Channels are **advertised** via a gossip protocol

Routing Properties

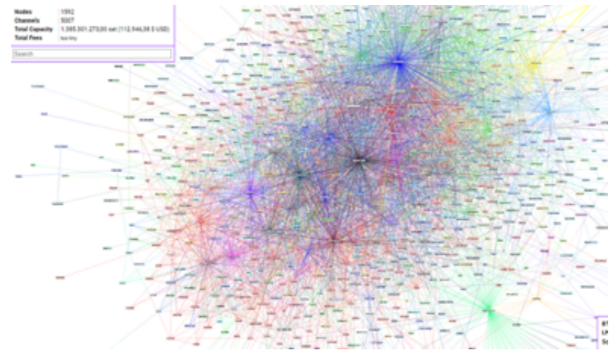


- **Scalable**:
  The routing algorithm should remain effective and efficient for large-scale PCN and high transaction rates.

- **Effectiveness**:
  Given the network topology, the routing algorithm should find the path that will likely be successful.

- **Efficiency**:
  The overhead of path discovery should be low in terms of communication, latency, and computation.

Routing Approaches



Topology is known.

Source-routing

Sender computes best path, during the transfer, each hop is told the identity of the next hop.

LIGHTNING

Topology is **not** known.

Per-hop routing

Every hop is aware of the final destination and they chose the next hop.

Silent Whispers by *Roos et al.*

Routing and Privacy

- **Value-Privacy**:
  Find paths without revealing the value of a transfer.

- **Transfer-Privacy**:
  Sender and receiver privacy should hold.

Channel Networks Summary

Theory (Avarikioti et al.) and practice (Lightning) suggest star-topologies.

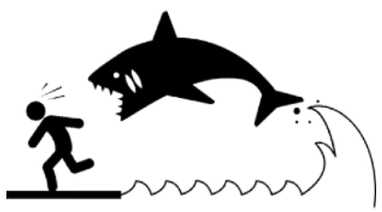
The good, bad and ugly of Payment Channels

Collateral for each hop

No direct connection needed

Decentralized, limited censorship

On-chain channel establishment



Wormhole attacks

Optimistically fast and cheap

Channel Networks
Trusted Execution
Environments

PCN based on Trusted Execution Environments

- **TEEs**:
  Efficient and fast payments among peers

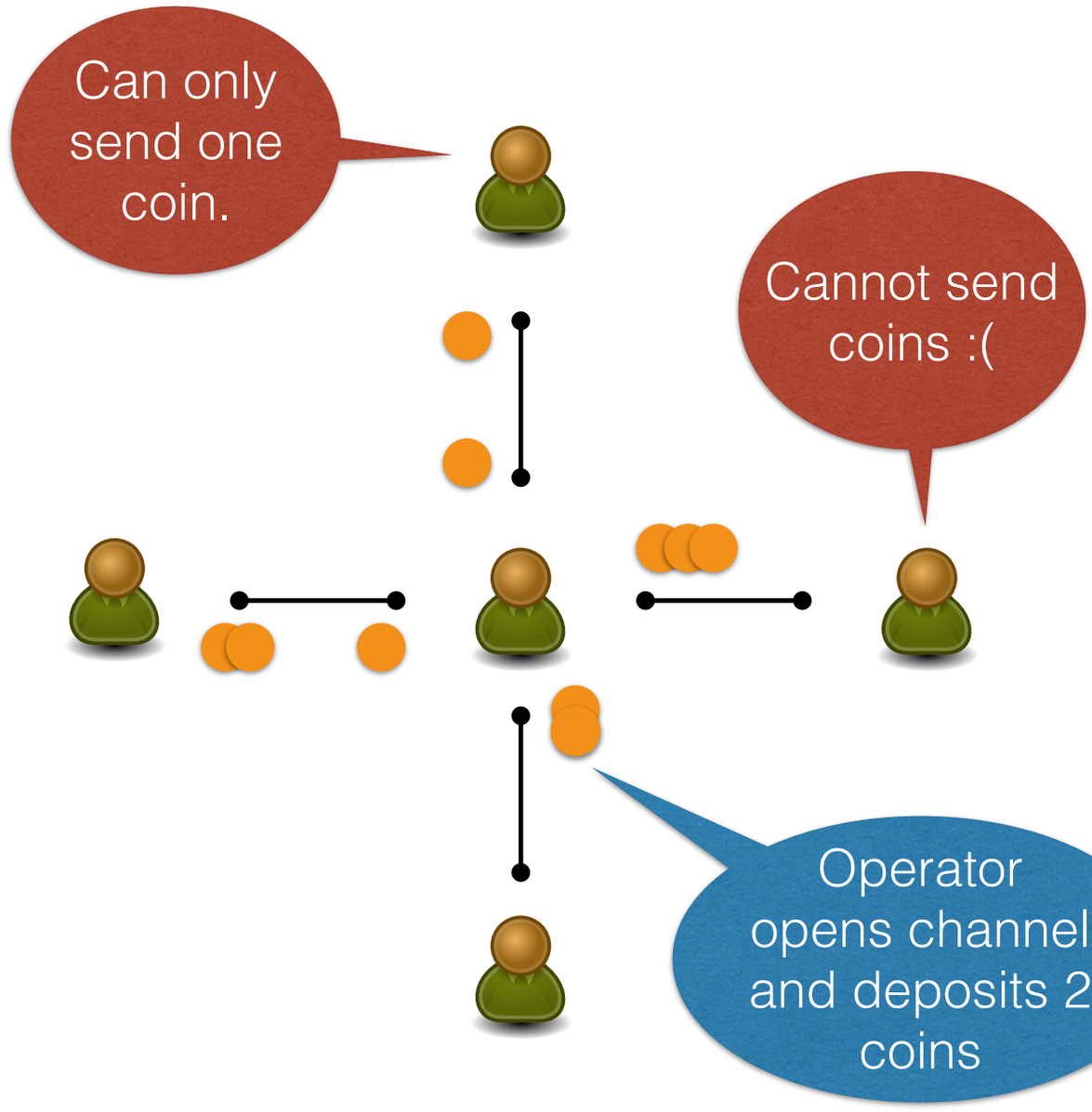
- **Privacy**:
  Intermediary nodes are not visible outside the enclave

- **Offline Payments**:
  Peers can remain offline!

- **Examples**:
  Teechan, Teechain

Payment Channel Hubs

Payment Channel Hub



Big pot of money

Expensive/slow on-chain transaction

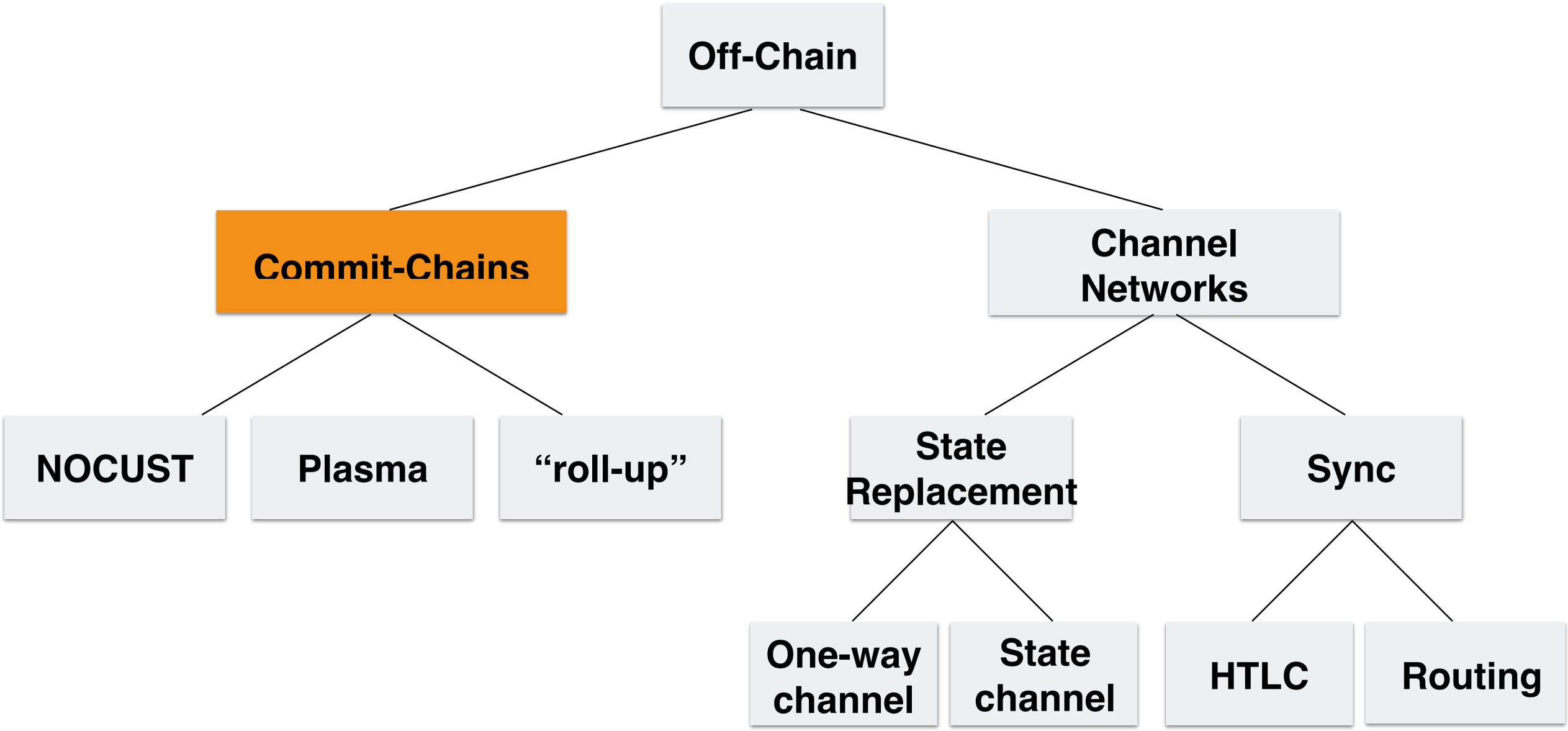
Payment Channel Hubs are great
for instant finality and no trust.

But expensive to run..

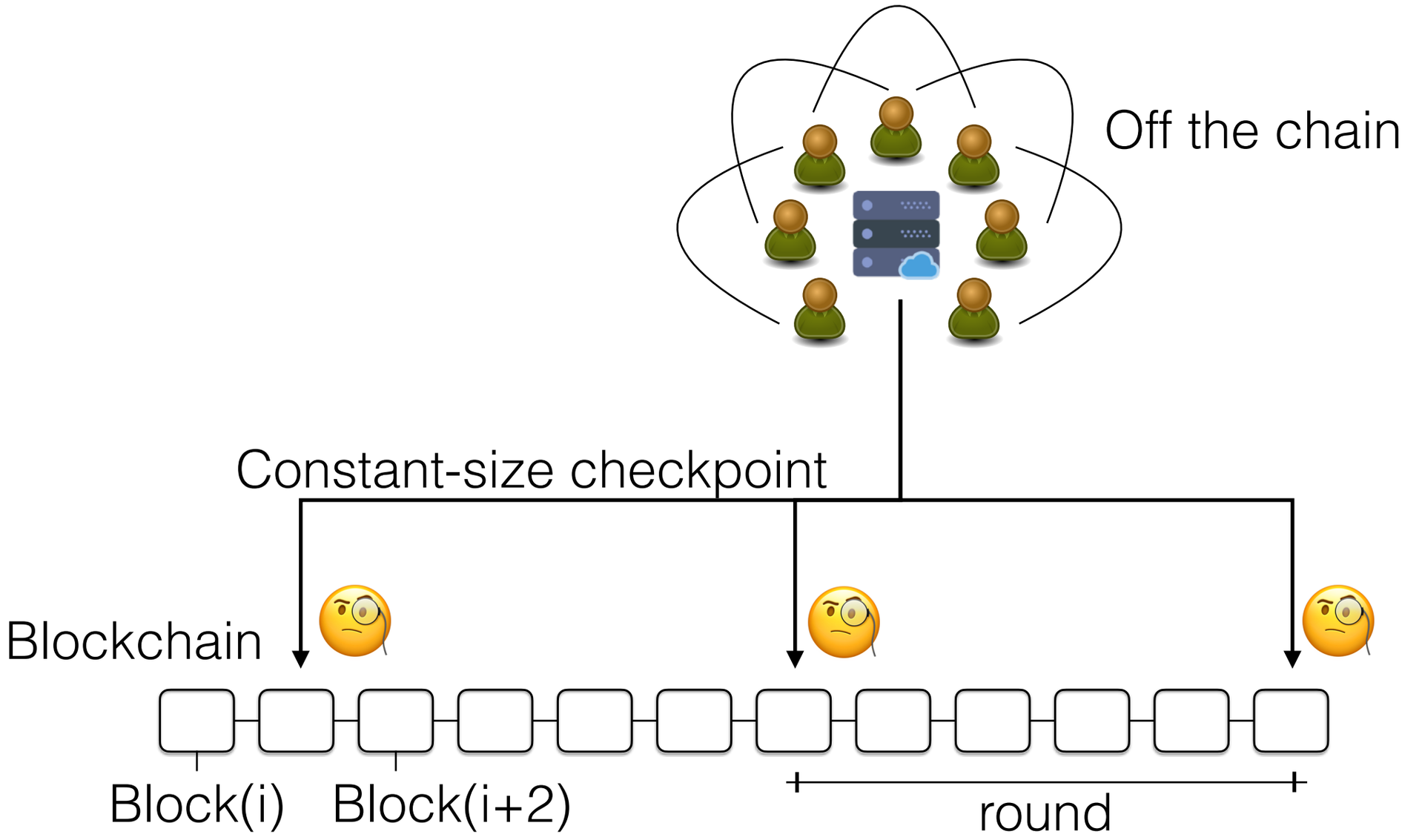
Can we do better with eventual finality?

Commit-Chains

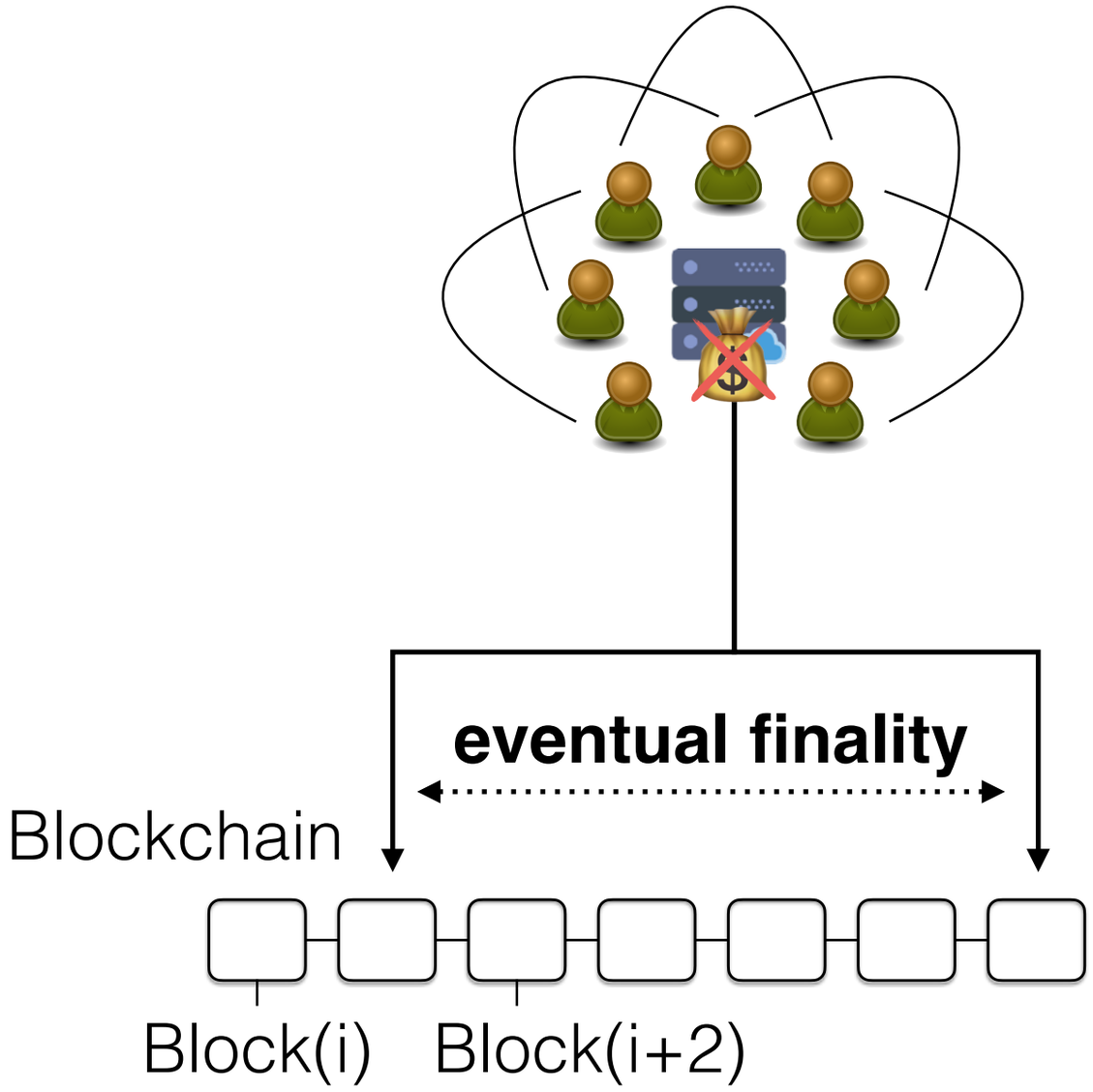
Which off-chain solution?



Commit-Chains



Without Collateral —> Eventual Finality

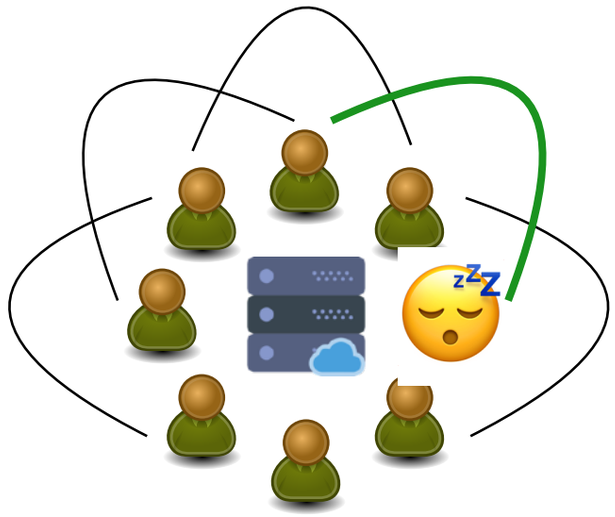


No collateral by operator

Recipient should wait..

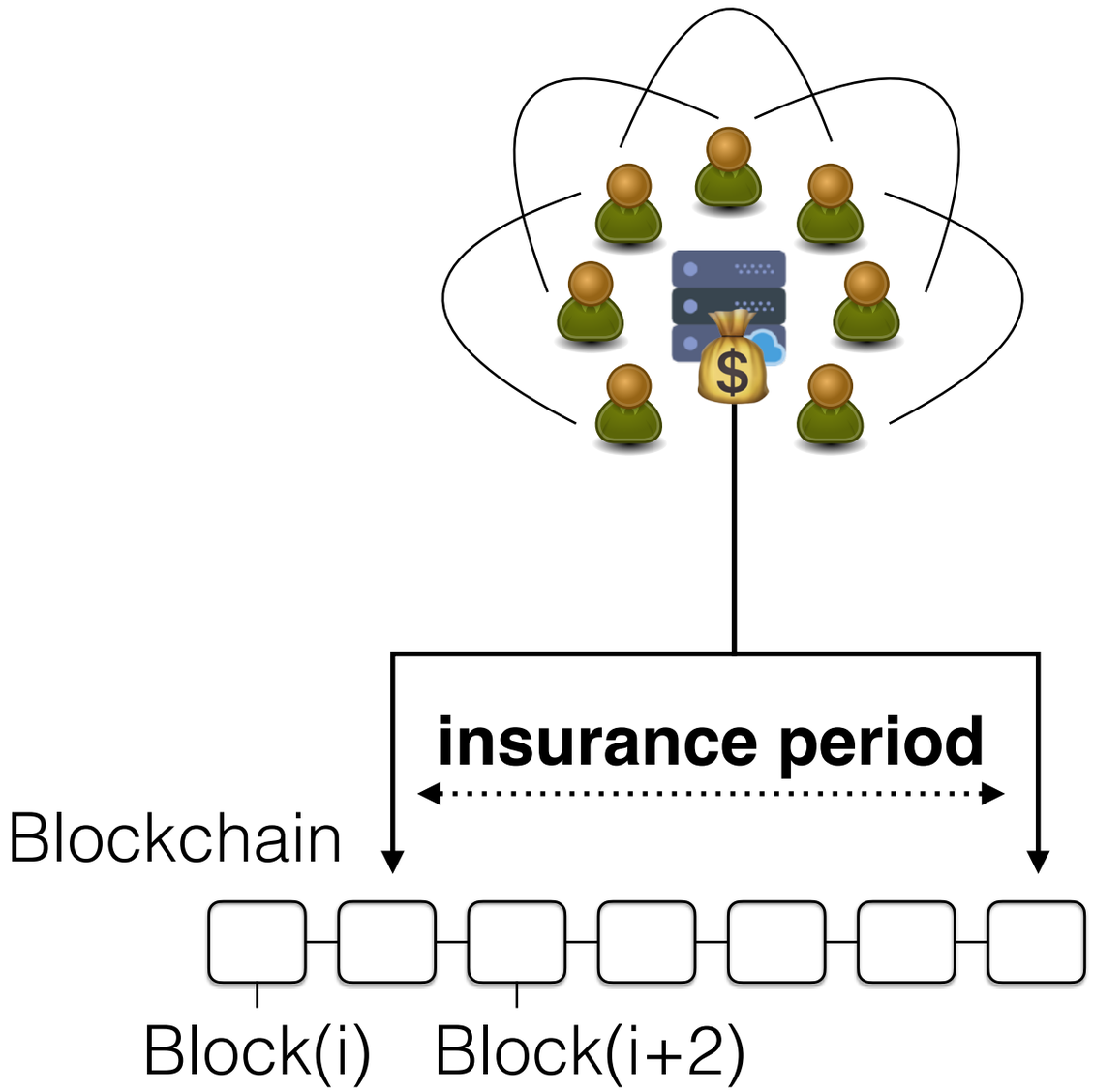
Receive TX while offline

Off the chain transactions



Like on-chain transactions..

With collateral —> Instant finality



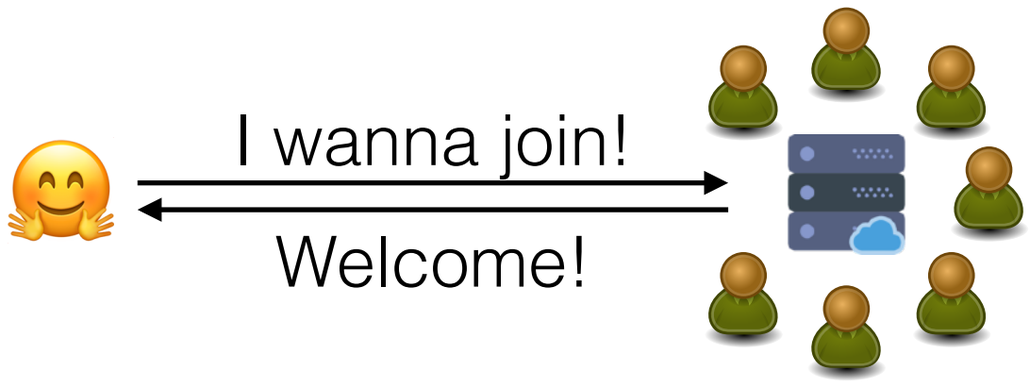
Insurance pool

Collateral re-usable each round

TX can be accepted instantly

Collateral allocation O(1) for all users

Join without on-chain Transaction



- Instant 🚀
- CO_2 friendly (Zero gas costs) 

Wait.. a centralized Operator

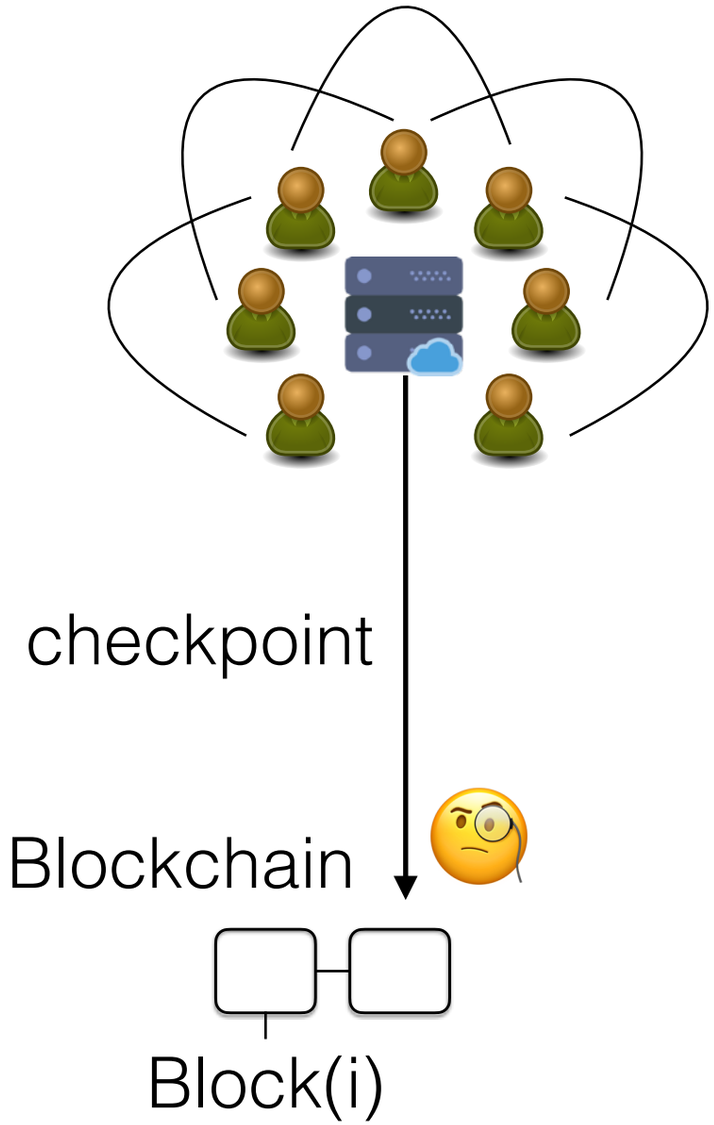
..but untrusted and non-custodial!

Cannot

- steal coins
- double spend coins
- mint new coins
  (if users follow the protocol..)

Commit-Chains
Plasma Cash

Plasma (Cash)

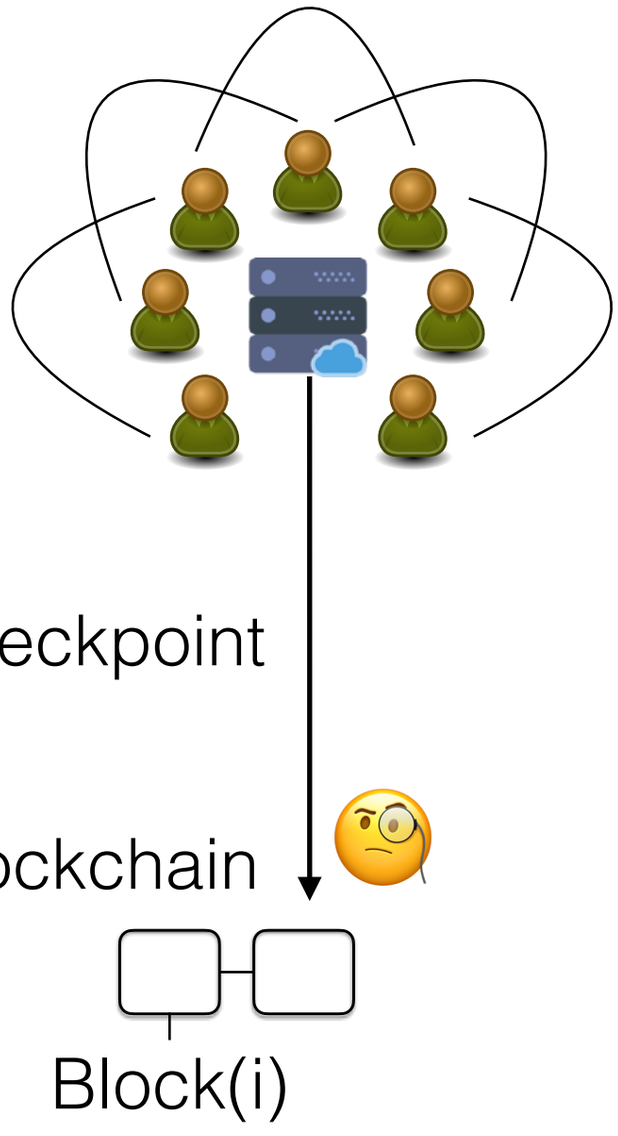


- **Non-Fungible Coins**:
  Deposit a coin -> Serial Number Coin

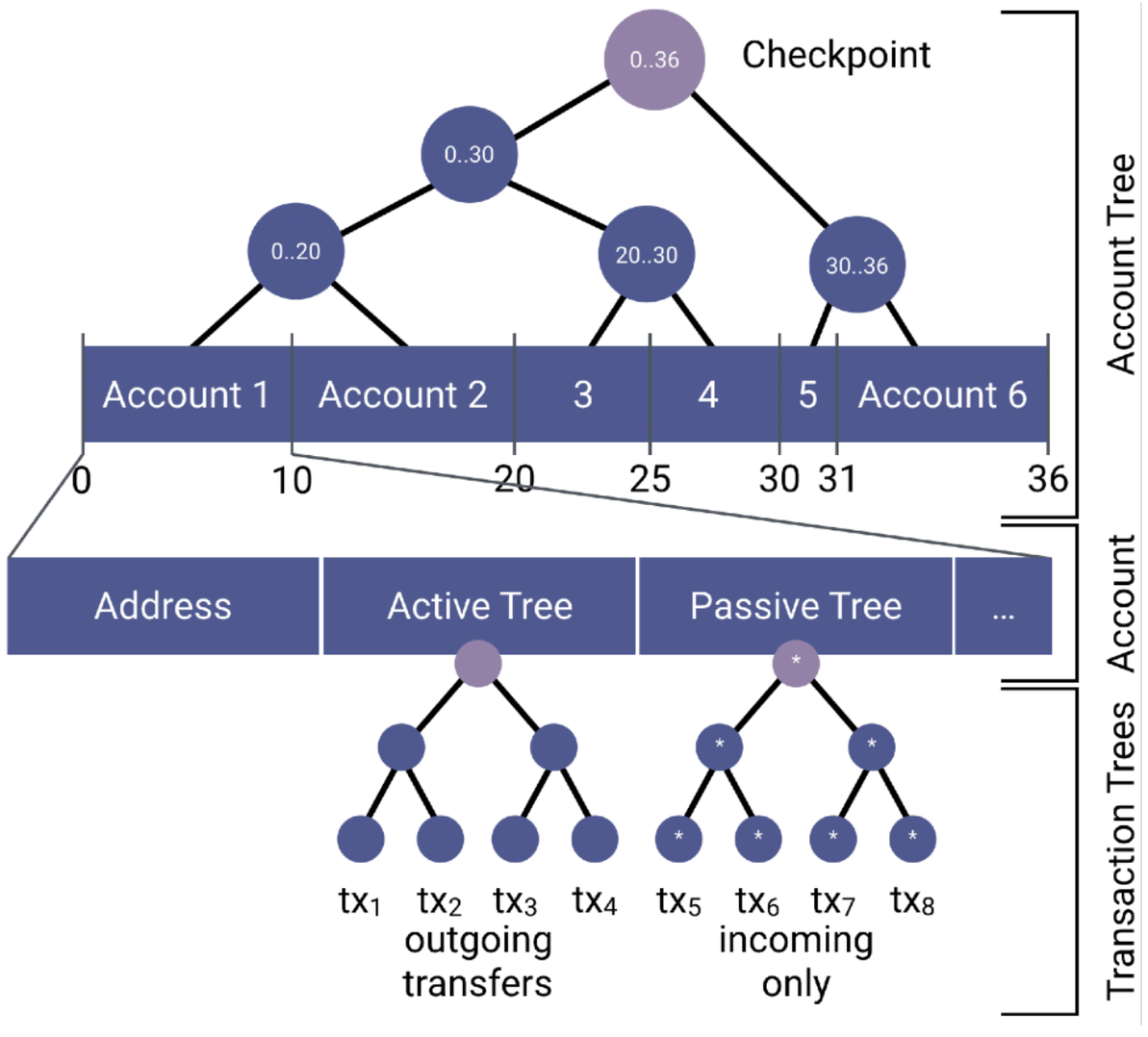
- **Merkle Tree**:
  Every leaf is a coin.

Commit-Chains
NOCUST

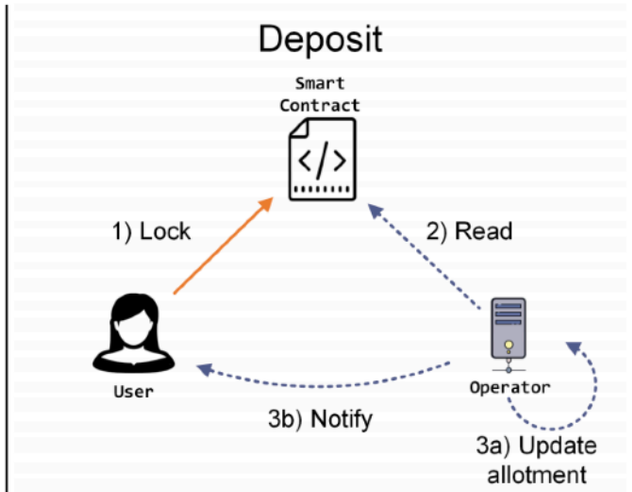
NOCUST

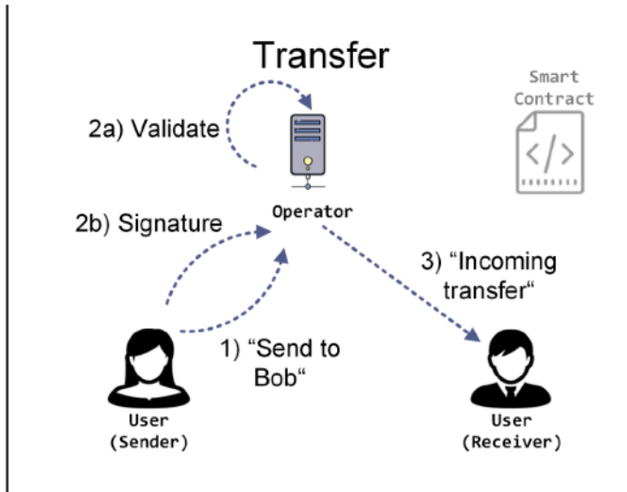


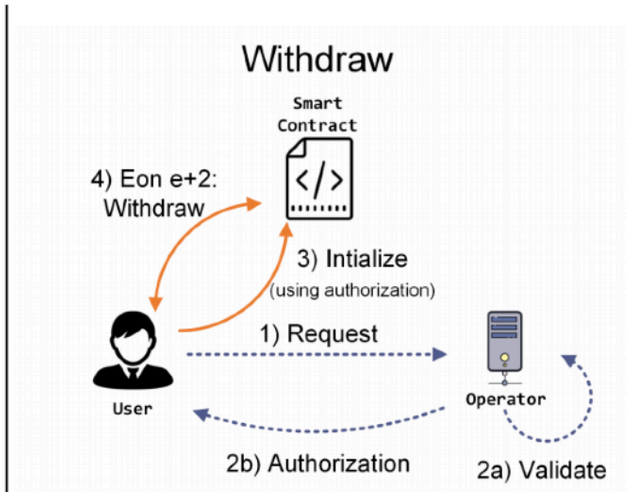
Fungible Coins:
Deposit a coin ->
Balance credit in a Merkle Tree

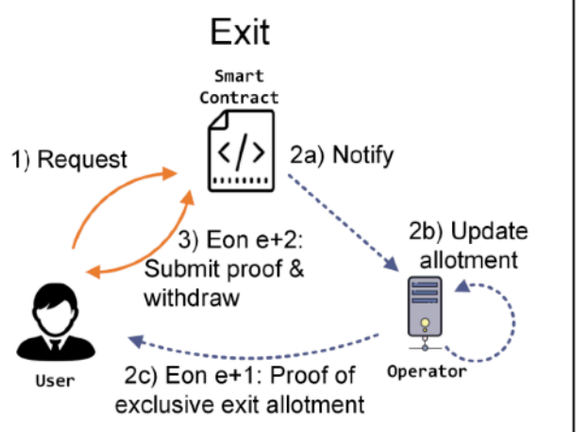


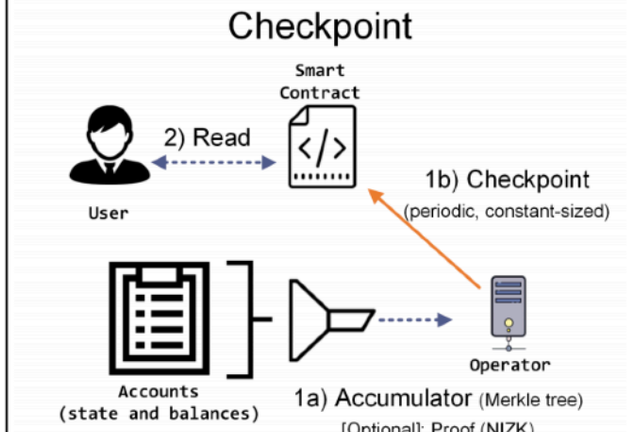
NOCUST

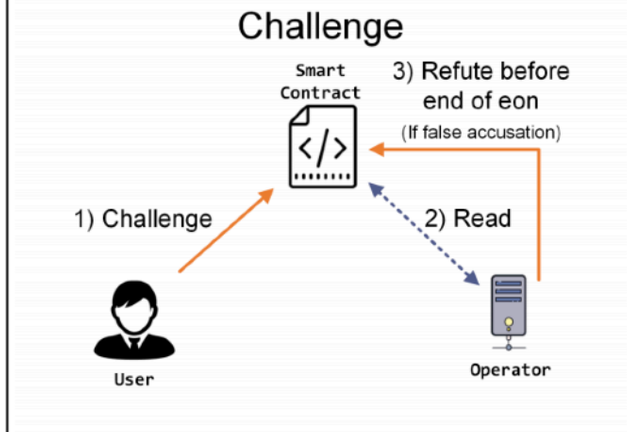


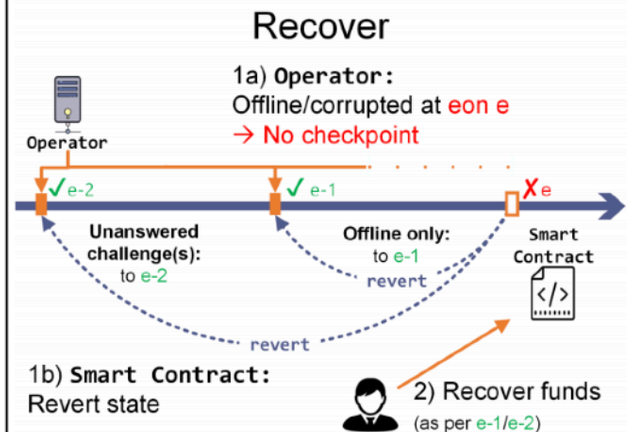






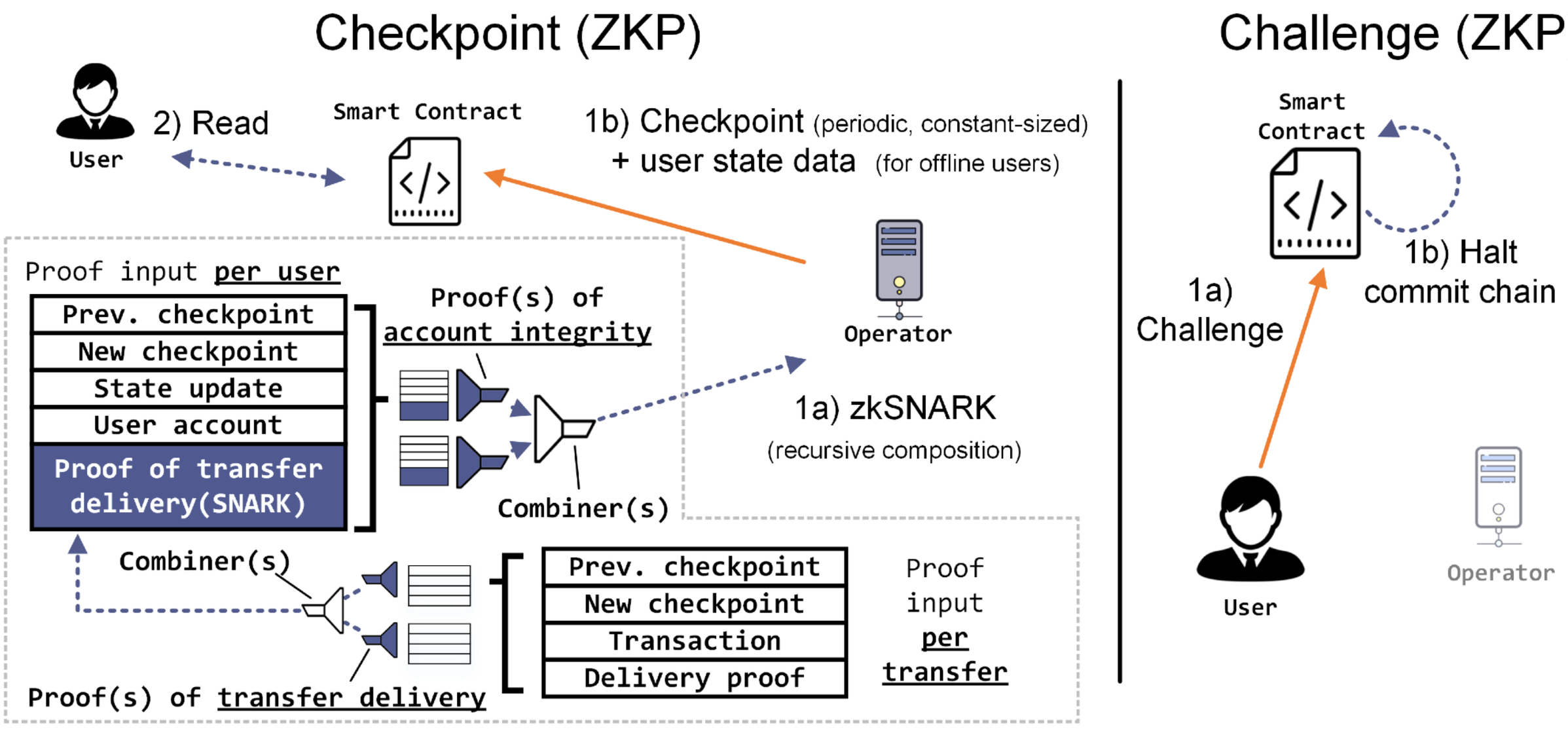






https://eprint.iacr.org/2018/642.pdf

NOCUST ZKP - Adding Succinct Proofs



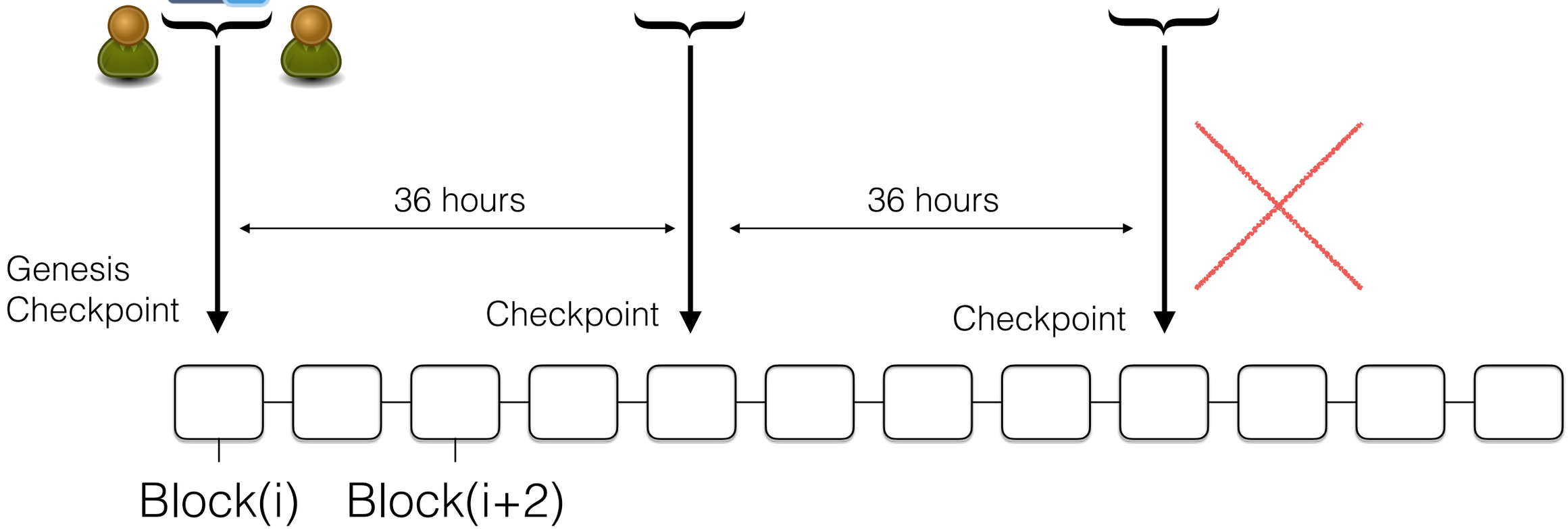
Commit-Chains
NOCUST Security

NOCUST operator disappears
… just after checkpoint

😊 **Transaction 1..200 are committed on-chain, can be withdrawn anytime**

Transactions 1..100

Transactions 101..200

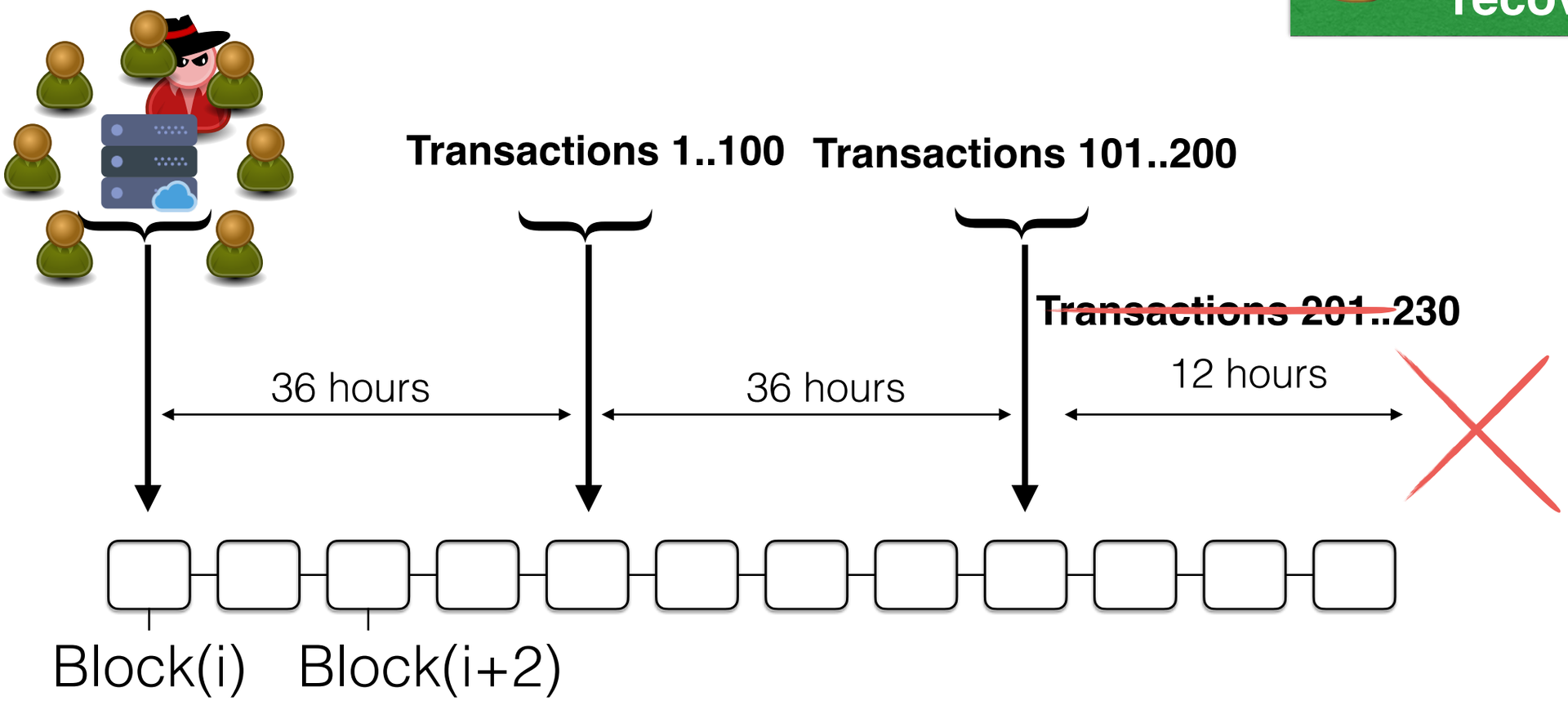


NOCUST server disappears … after a few transactions

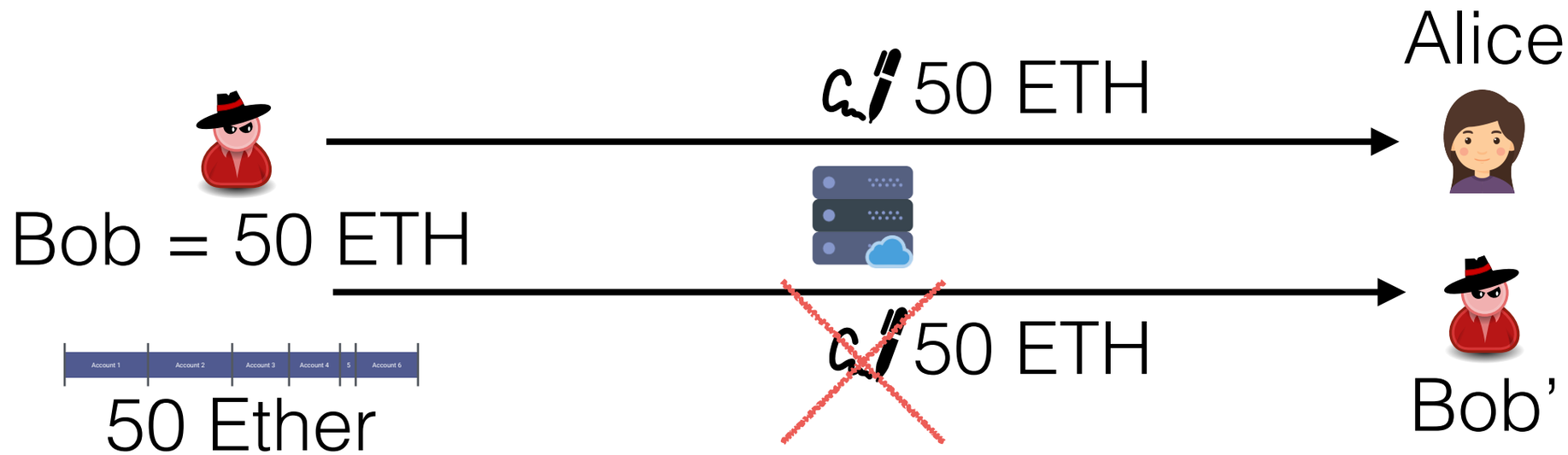
😊 **Transaction 1..200 are safe**

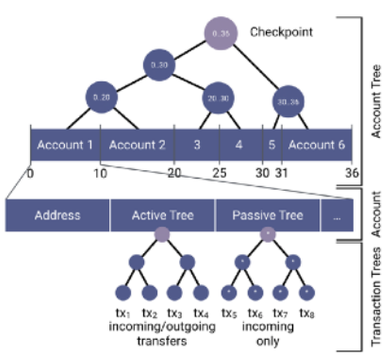
😫 **Transaction 201..230 are not committed on chain!**

💰 **Insurance Pool recovers tx** 😊

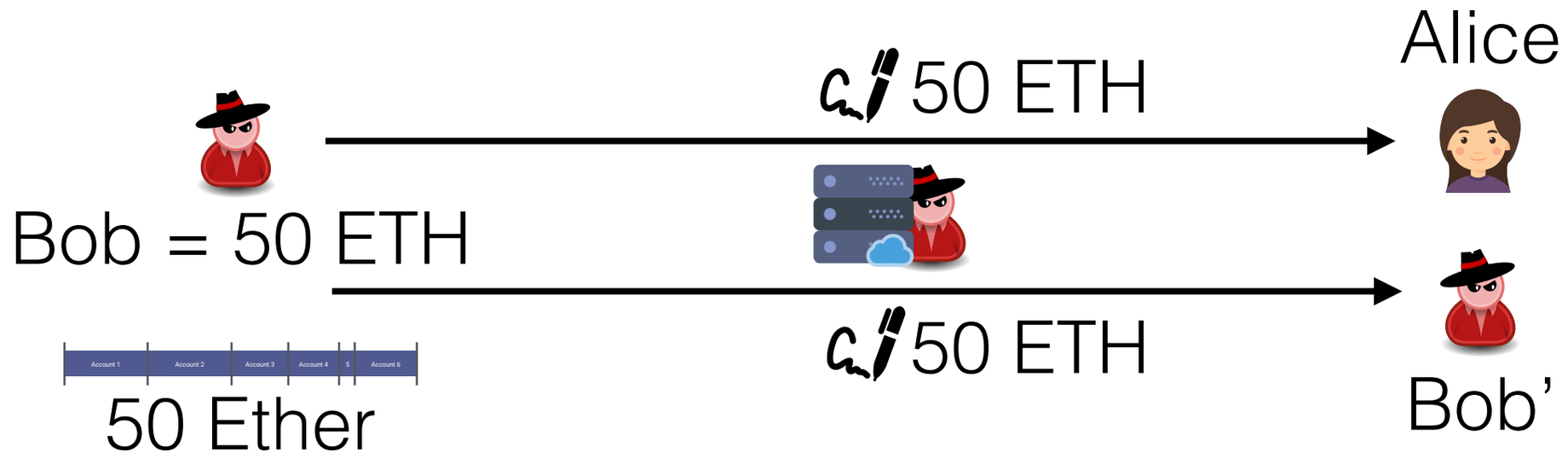


Users to attempt double-spending





NOCUST server colludes with client to attempt double-spending



Attempt to create coins

50 Ether != 100 Ether

Operator is challenged !

Attempt to steal coins

50 Ether of Alice are lost

Operator is challenged !

Commit-Chains Summary

NOCUST

Off-chain ledger state:
user balances

Fungible payments

Slower delayed finality

Lightweight clients

Instant finality support

Atomic Off-chain Swap (TEX)

Regular online presence to watch malicious state changes

NOCUST-ZKP

- trusted setup for ZK

+ users without incoming transaction can stay offline

+ non-interactive challenge

Plasma (Cash)

Off-chain ledger state:
coin serial number

Non-Fungible payments

Rapid delayed finality

Large amounts of history data

No instant finality support

No known feasibility for swaps

Regular online presence to watch malicious withdrawals

The good, bad and ugly of Commit Chains

Recipient can be offline to receive a transaction

Limited censorship resistance

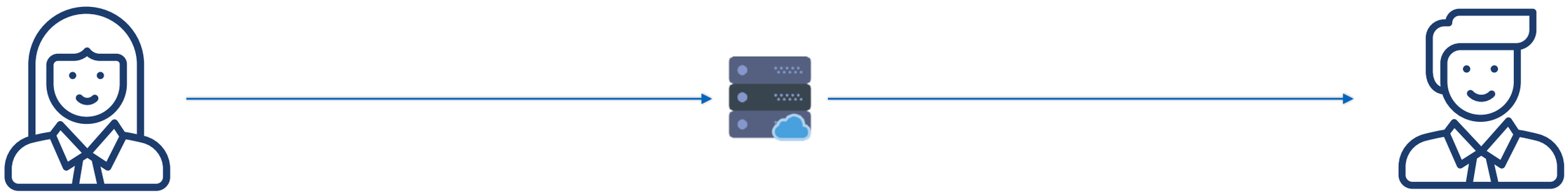
No decentralization

No collateral for delayed finality

Instant and free onboarding

Data availability challenges

What if we have a blockchain that scales significantly (wrt. latency and volume)?



2nd Factor and Location Verification:
Do you really want to spend these coins?
-> Get notified and act upon misbehavior.

Privacy:
Off-Chain transactions are not broadcast, **may** increase privacy
Privacy is lost towards commit-chain operators.