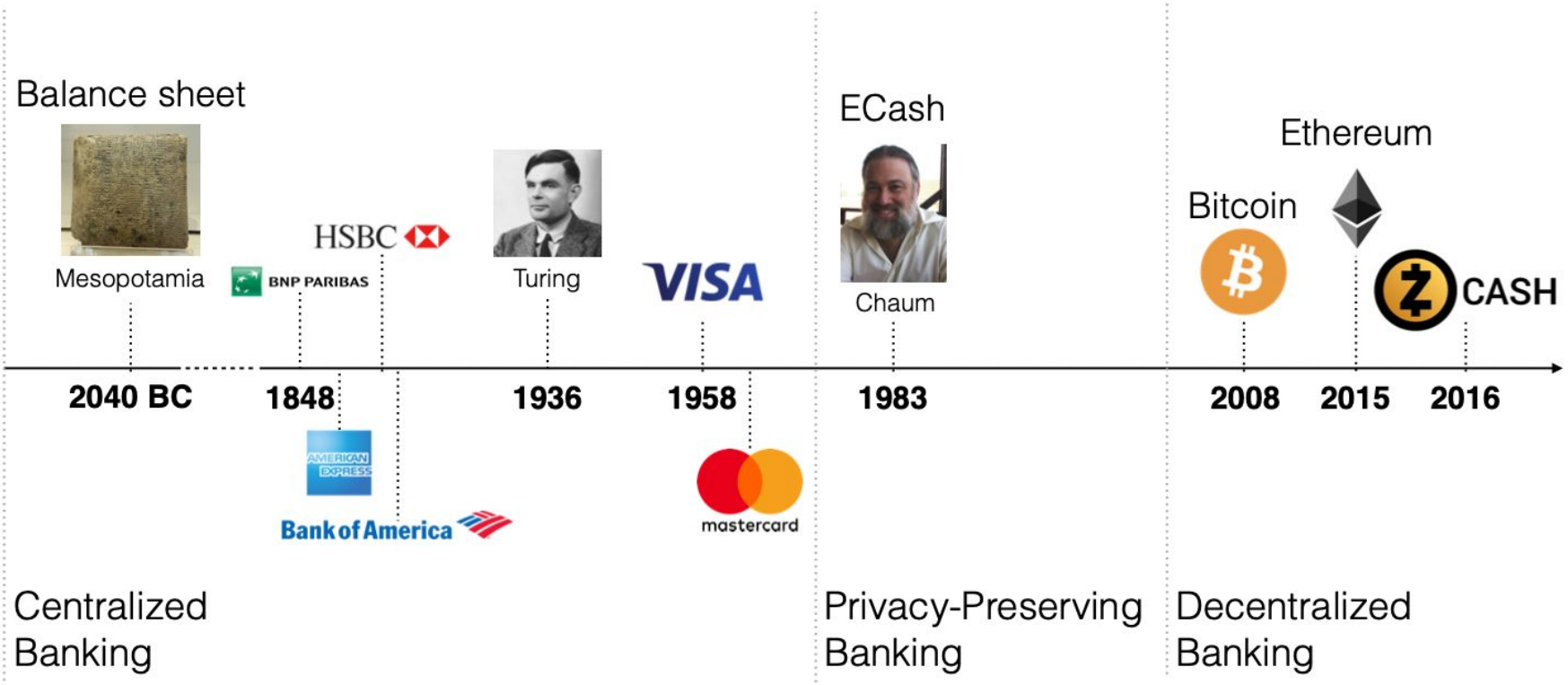
Building blocks

- 2040 BC

Balance sheet - Mesopotamia

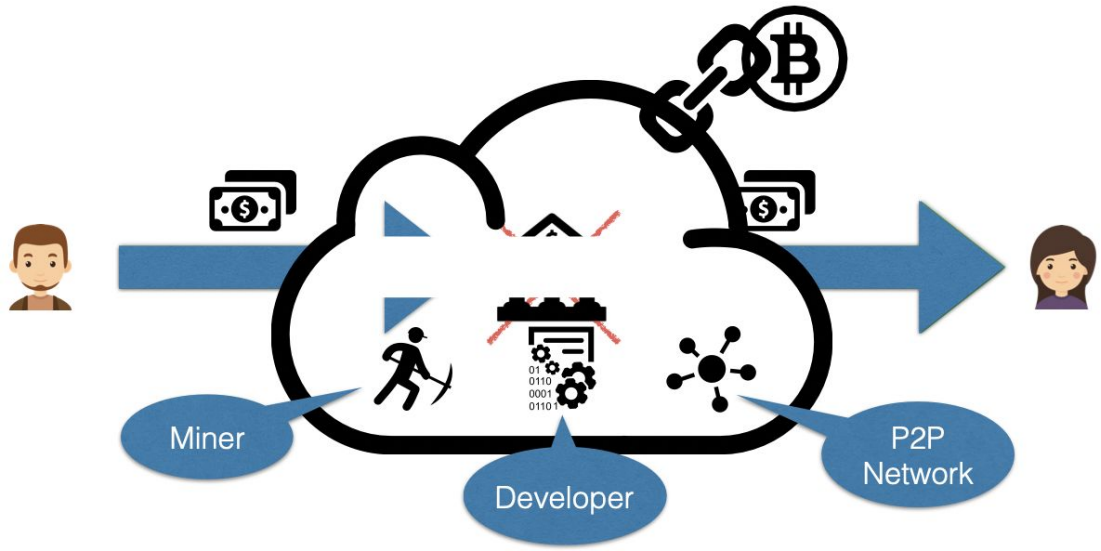
Building blocks

- Evolution of accounting



Building blocks

- From Centralized to Decentralized payment systems



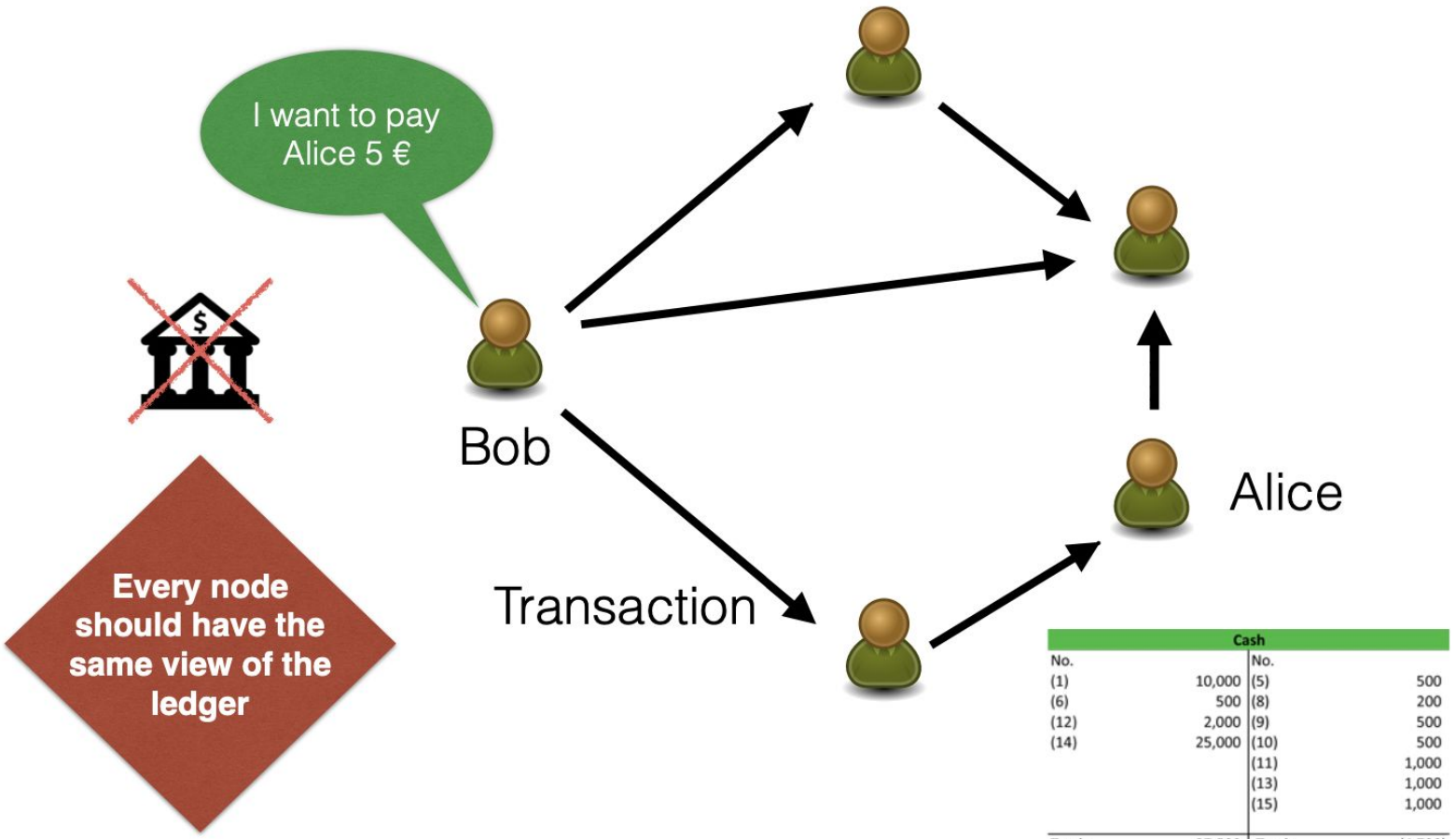
  - How to perform secure **decentralized** payments?
  - How to exchange **privacy-preserving** payments?
  - How to make decentralized systems **efficient**?

Building blocks

- Digital payment systems
  - With a centralized entity 
    - Banks
    - Digital signatures are main double-spending resistance mechanism
  - Examples
    - Pepper Micropayments [Rivest]
    - ECash [David Chaum]
      - Privacy preserving

Building blocks

- Decentralized digital payments



Cash			
No.		No.	
(1)	10,000	(5)	500
(6)	500	(8)	200
(12)	2,000	(9)	500
(14)	25,000	(10)	500
		(11)	1,000
		(13)	1,000
		(15)	1,000
Totals	37,500	Totals	(4,700)

Building blocks

- A Blockchain is a (distributed) database

Building blocks

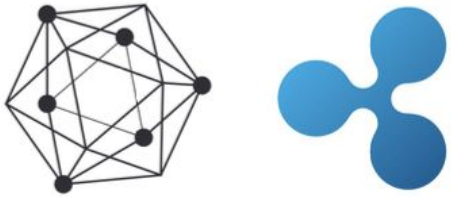
- Blockchain participants
  - Writer

  - Reader

Building blocks

- Blockchain != Blockchain
  - “Open and Decentralised” Blockchains
    - Bitcoin
    - Ethereum
    - ...

  - Permission-based Blockchains
    - Hyperledger
    - Ripple
    - Stellar
    - ...



Building blocks

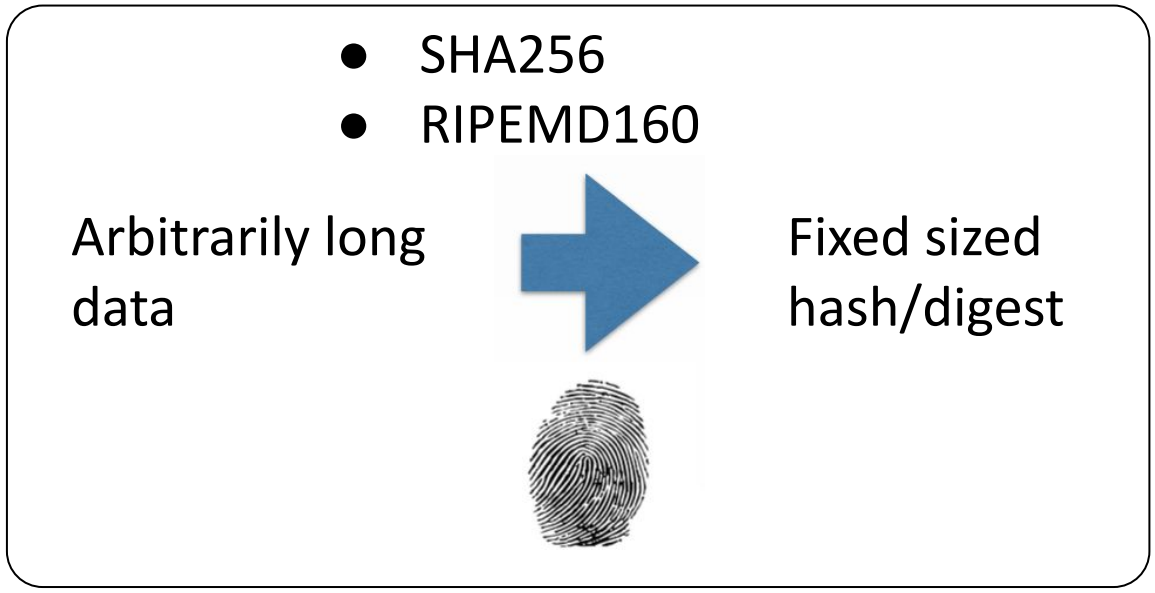
- “Open and Decentralised” Blockchains

Building blocks

- Bitcoin
  - First introduced in 2009
    - By a pseudonym Satoshi Nakamoto
  - Peer-to-peer decentralized currency
  - No trusted third parties
  - It's Blockchain is Distributed DB
    - Transactions
    - Blocks
  - 0 < Balance < 21 Million Bitcoin
    - 1 Satoshi = 10^{-8} Bitcoin

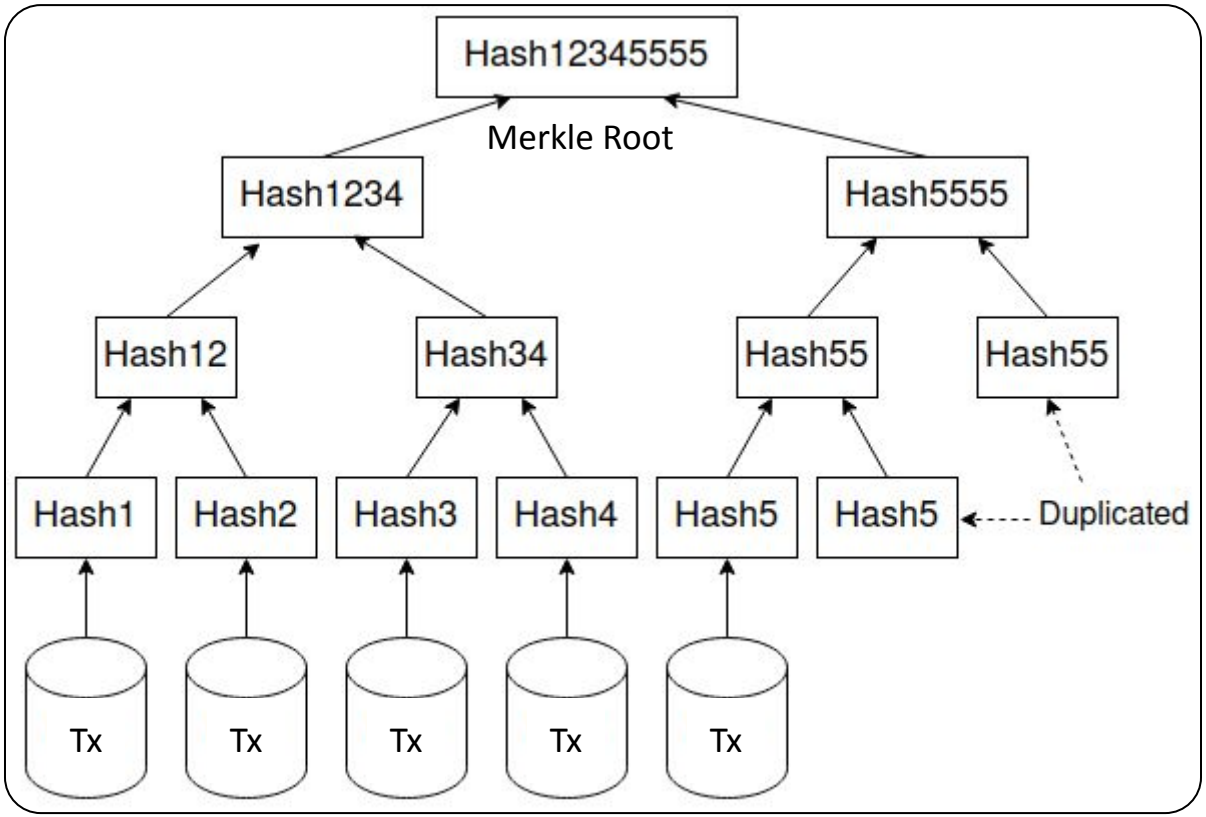
Building blocks

- Data types
  - Cryptographic Hash Functions



Building blocks

- Data types
  - Merkle trees



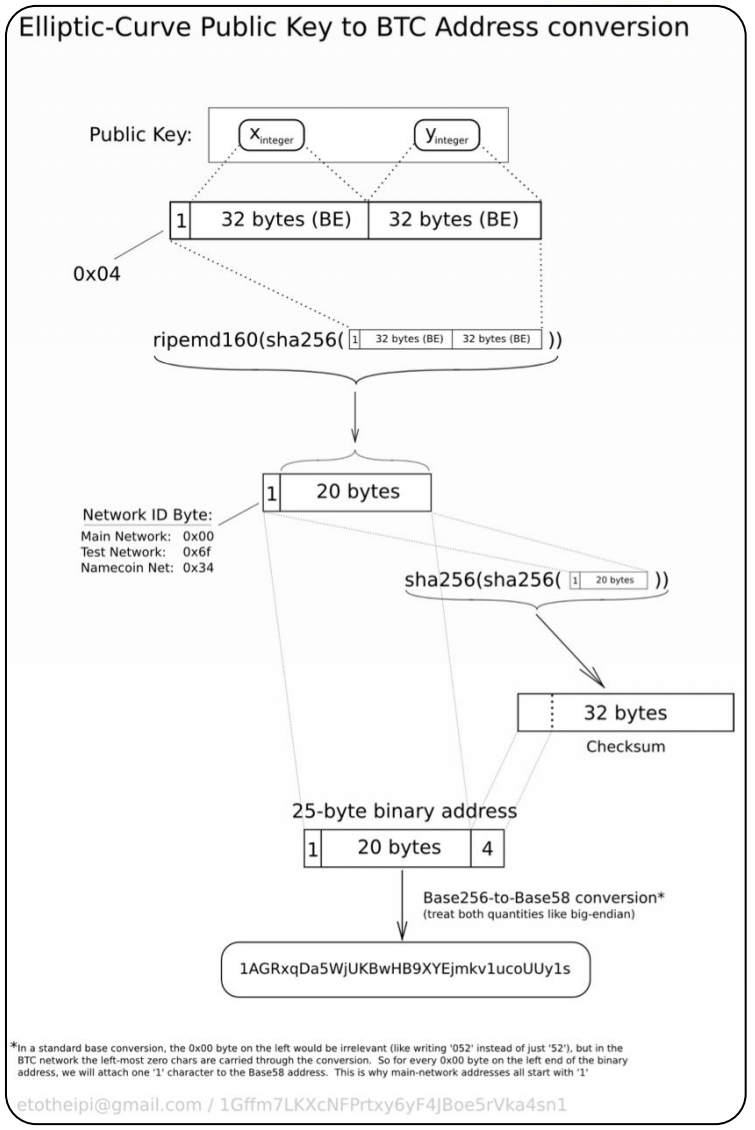
Building blocks

- Data types
  - Elliptic Curve Signature Algorithm (ECDSA)

- ECDSA (secp256k1 curve) is used to
  - Sign transactions
  - Verify the signature of transactions
- Nothing in Bitcoin is encrypted

Building blocks

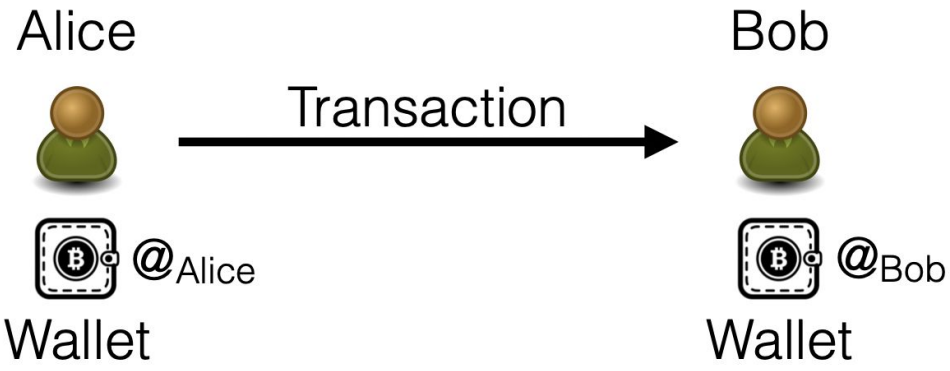
- Addresses
  - Unique identifier
  - Hash of a public key

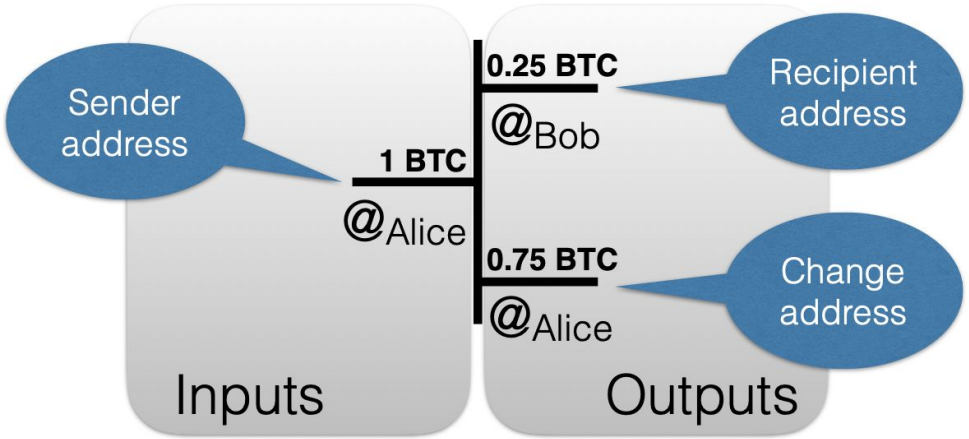


Type	Version prefix (hex)	Base-58 result's prefix
P2PKH address	0x00	1
P2SH Address	0x05	3
Bitcoin Testnet Address	0x6F	m or n
Private Key WIF	0x80	5, K, or L
BIP32 Extended Public Key	0x0488B21E	xpub
Bech32 (SegWit)	0x0303000203 (for checksum only)	bc1

Building blocks

- Transactions





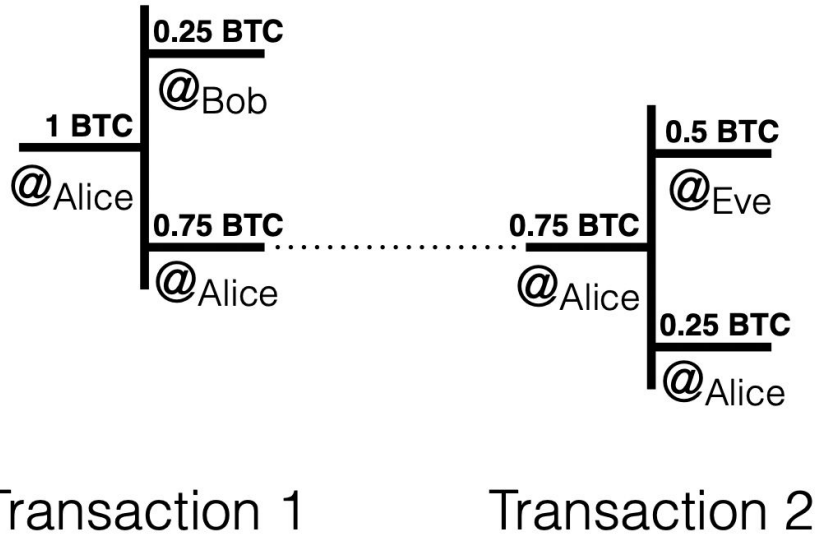
Transaction Fees

$$\sum inputs \geq \sum outputs$$

Difference are fees

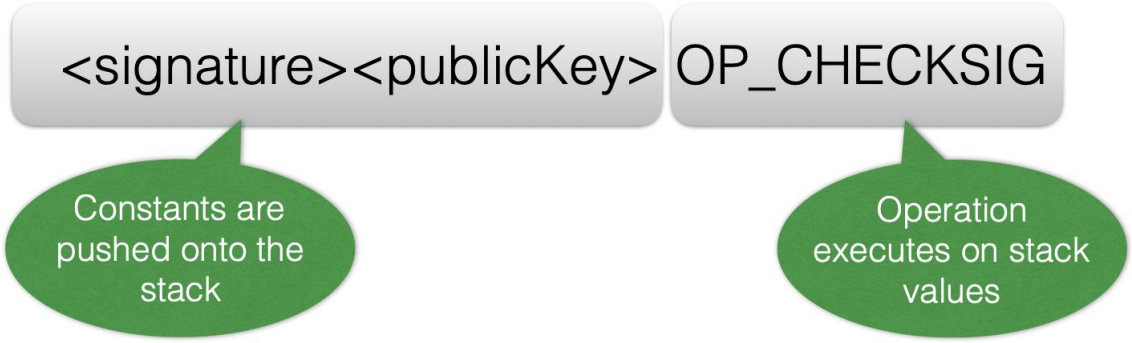
Building blocks

- Transactions



Building blocks

- Script
  - Stack based programming language
    - Turing incomplete
  - If evals to *true* —> Bitcoin transaction is valid
  - Many opcodes
  - Execution time is critical to prevent DoS attacks
  - E.g.,



Building blocks

- Transactions in Ethereum
  - Transaction with signature which modifies an **account**

type	from	sig	nonce	to	data	value	gaslimit	gasprice

Building blocks

- Ethereum Virtual Machine
  - EVM Features
    - Stack of max depth of 1024
    - 32-byte words
    - Dedicated crypto opcodes
    - SHA-3
    - Big num multiply
    - GF-256 operators

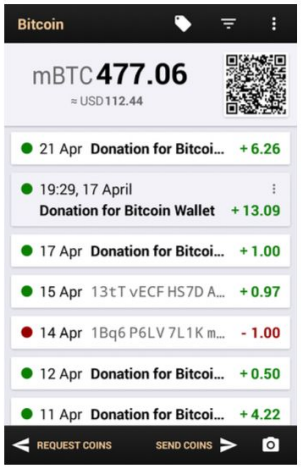
EVM code

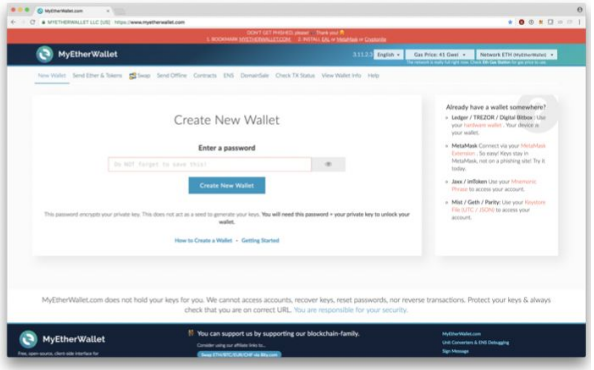
```
.code
  PUSH 60             contract Ballot {\n
    struct...
  PUSH 40             contract Ballot {\n
    struct...
  MSTORE              contract Ballot {\n
    struct...
  CALLVALUE                function Ballot(uint8 _numProp...
  ISZERO              function Ballot(uint8 _numProp...
  PUSH [tag] 1             function Ballot(uint8 _numProp...
  JUMPI               function Ballot(uint8 _numProp...
  PUSH 0              function Ballot(uint8 _numProp...
  DUP1                function Ballot(uint8 _numProp...
  REVERT              function Ballot(uint8 _numProp...
tag 1            function Ballot(uint8 _numProp...
  JUMPDEST                 function Ballot(uint8 _numProp...
  PUSH 40             function Ballot(uint8 _numProp...
  MLOAD               function Ballot(uint8 _numProp...
  PUSH 20             function Ballot(uint8 _numProp...
  DUP1                function Ballot(uint8 _numProp...
  PUSHSIZE                 function Ballot(uint8 _numProp...
  DUP4                function Ballot(uint8 _numProp...
  CODECOPY                 function Ballot(uint8 _numProp...
  DUP2                function Ballot(uint8 _numProp...
  ADD                 function Ballot(uint8 _numProp...
  PUSH 40             function Ballot(uint8 _numProp...
  MSTORE              function Ballot(uint8 _numProp...
  DUP1                function Ballot(uint8 _numProp...
  DUP1                function Ballot(uint8 _numProp...
```

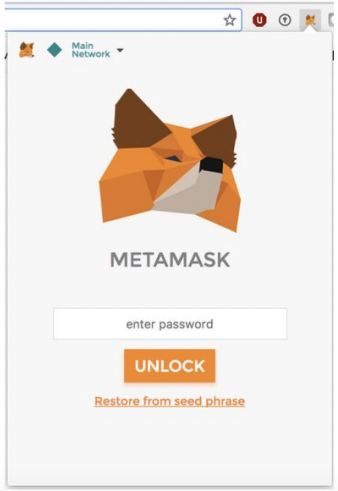
Joseph Bonneau, https://docs.google.com/presentation/d/1NZlzs-Hy3f7H2Hqh61BEc6wA19KD5UTirvLq_JBta18

Building blocks

- Wallets
  - Bitcoin Core
    - Full node, downloads all txs
  - Bitcoin Wallet
    - Android
  - Jaxx
    - Multi-chain wallet
  - Ledger Nano S







Building blocks

- What can you do with Crypto?
  - Trade with other people
  - Exchange to other crypto/FIAT
  - Buy something online
  - Lend
  - Donate
  - Build applications/games
  - ...

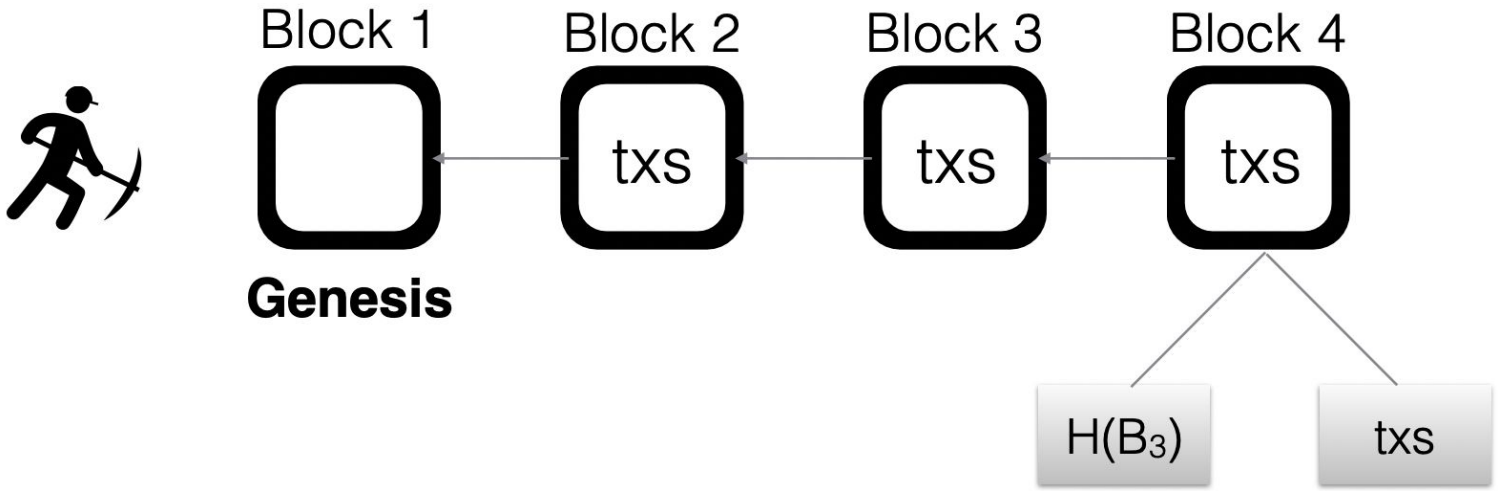
Building blocks

- Block

https://bitcoin.org/en/developer-reference#block-headers

Building blocks

- Blockchain

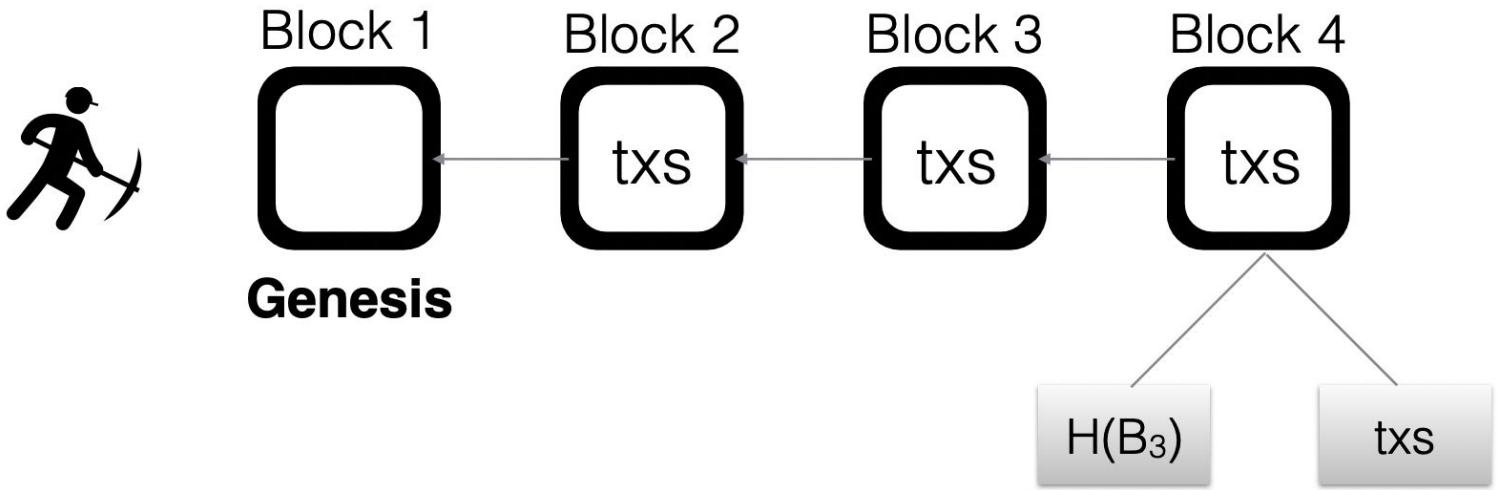


Building blocks

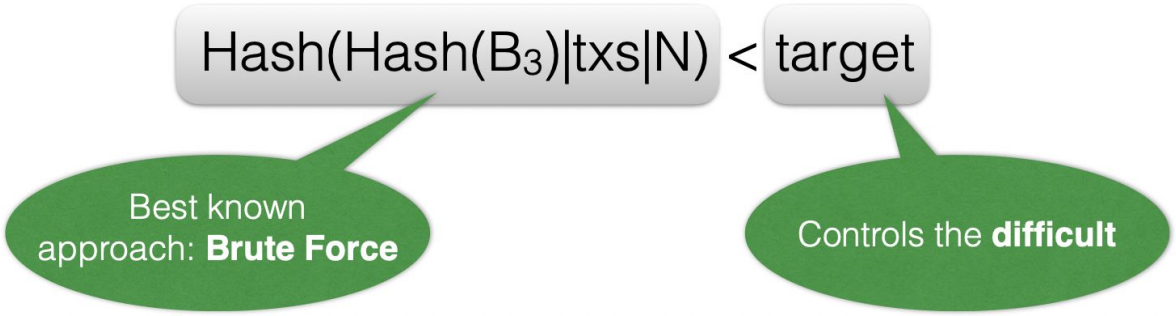
- Mining
  - Join the network, listen for transactions
    - Validate all incoming transactions
  - Listen for new blocks, build on valid blocks
  - Construct a block template
  - Find a nonce that validates the new block
  - Tell all the other miners
  - Receive reward

Building blocks

- Blockchain



  - Mining
    - Find Nonce N, such that



Building blocks

- Mining difficulty

$$Hash(Hash(B_3)|txs|N) < target = 0x\underline{000}**$$

~~Hash(Block_3 | merkle_root | 0xabca) = 0x03ef..~~

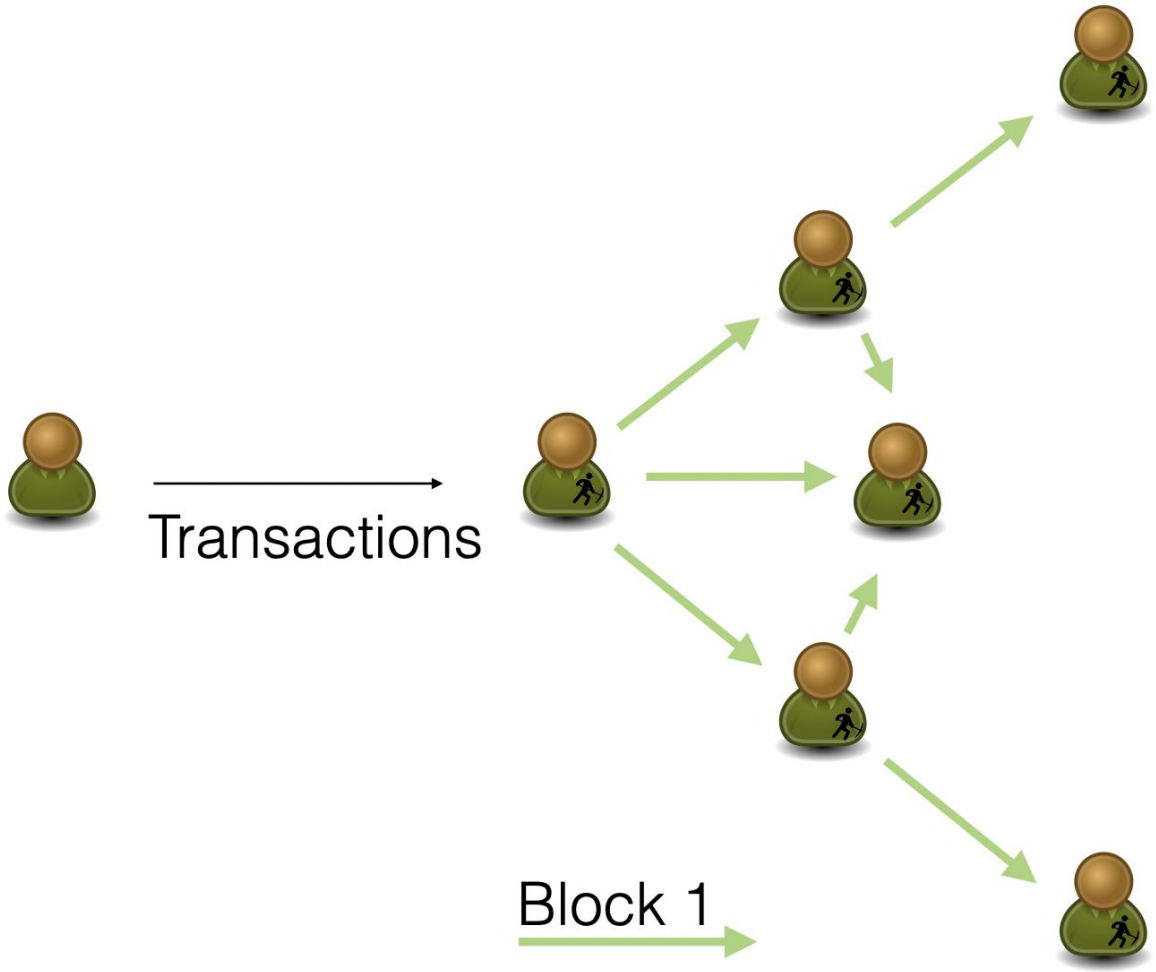
~~Hash(Block_3 | merkle_root | 0xabcb) = 0x12ef..~~

~~Hash(Block_3 | merkle_root | 0xabcc) = 0x20ef..~~

Hash(Block_3 | merkle_root | 0xabcd) = 0x000f..

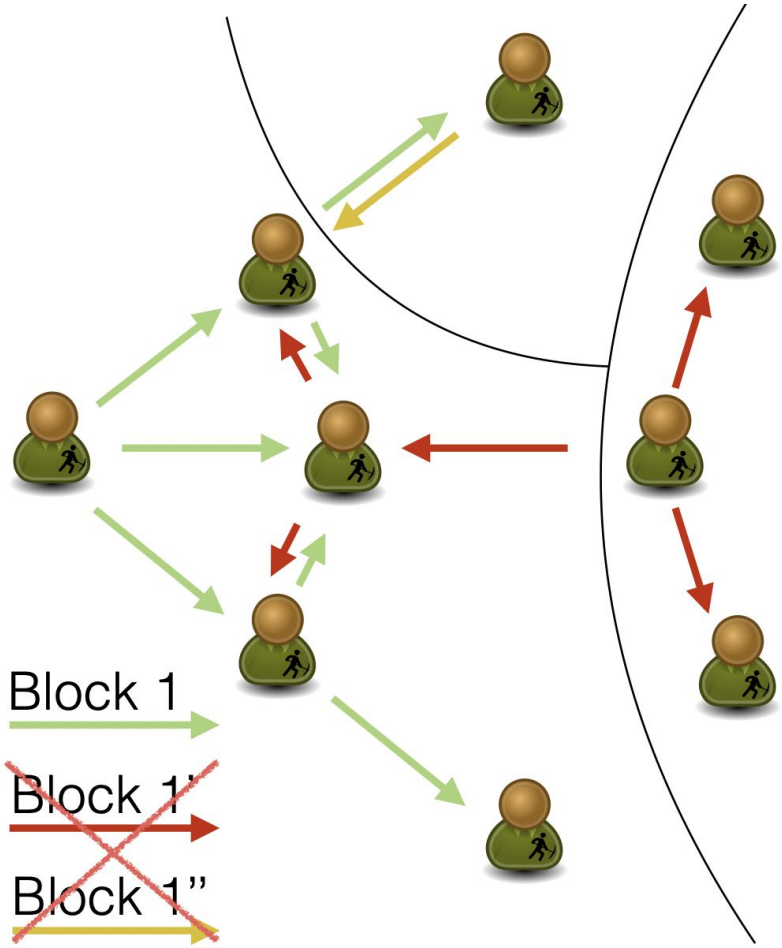
Building blocks

- Miners



Building blocks

- Mining and forks



- Network partition
- **Stale blocks = lost efforts**

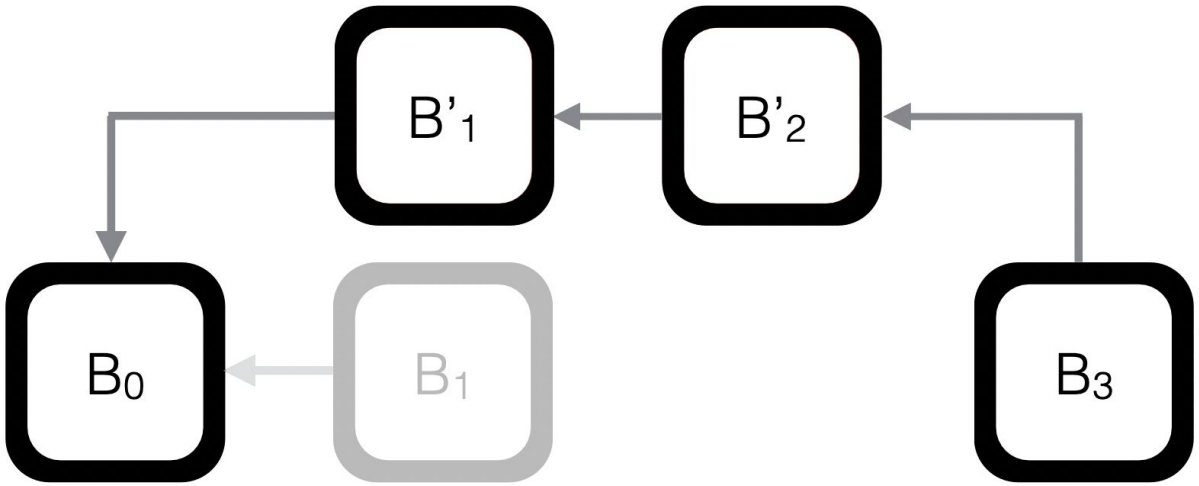
Selfish Mining

Denial of Service

Double Spending

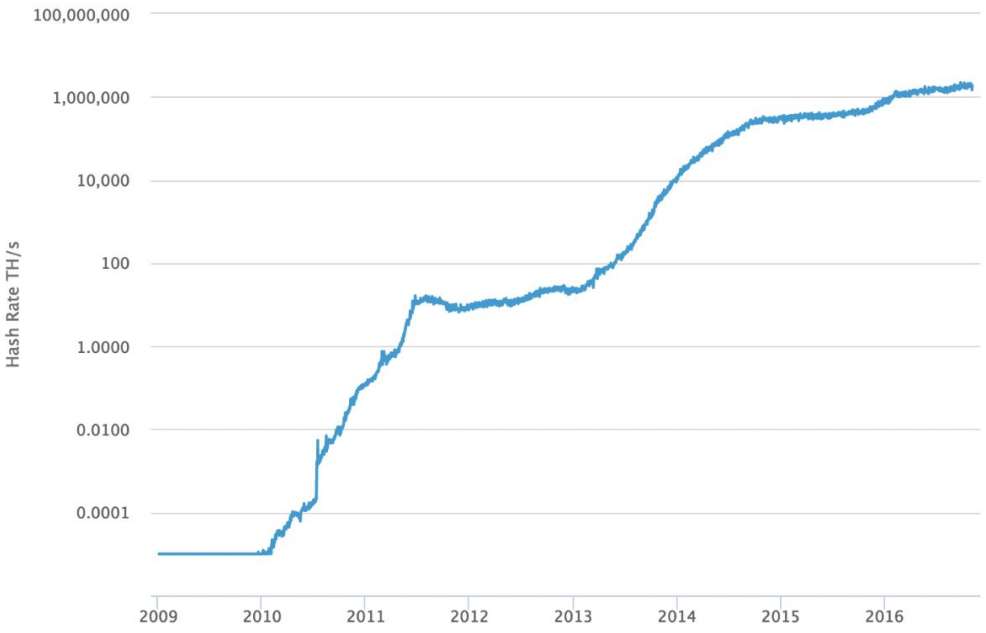
Building blocks

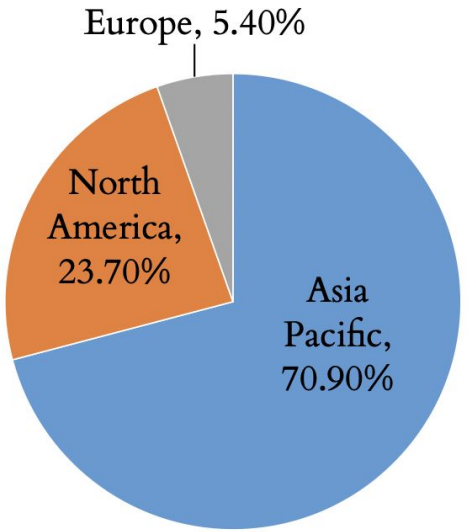
- Blockchain forks



Building blocks

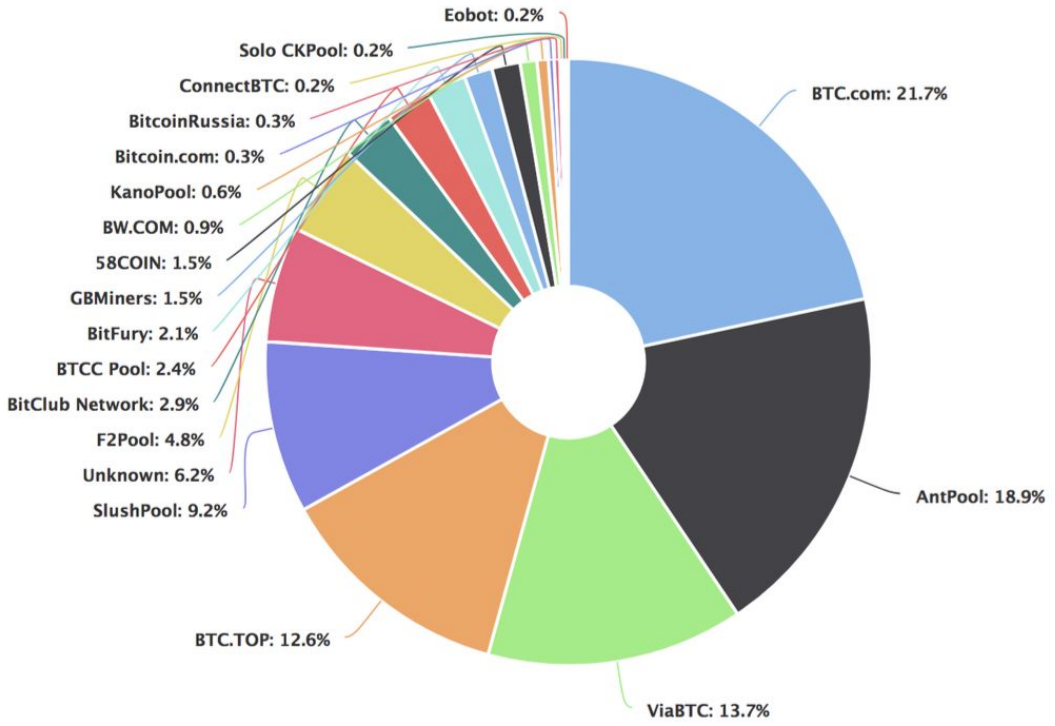
- Mining
  - From CPU to GPU to ASICs





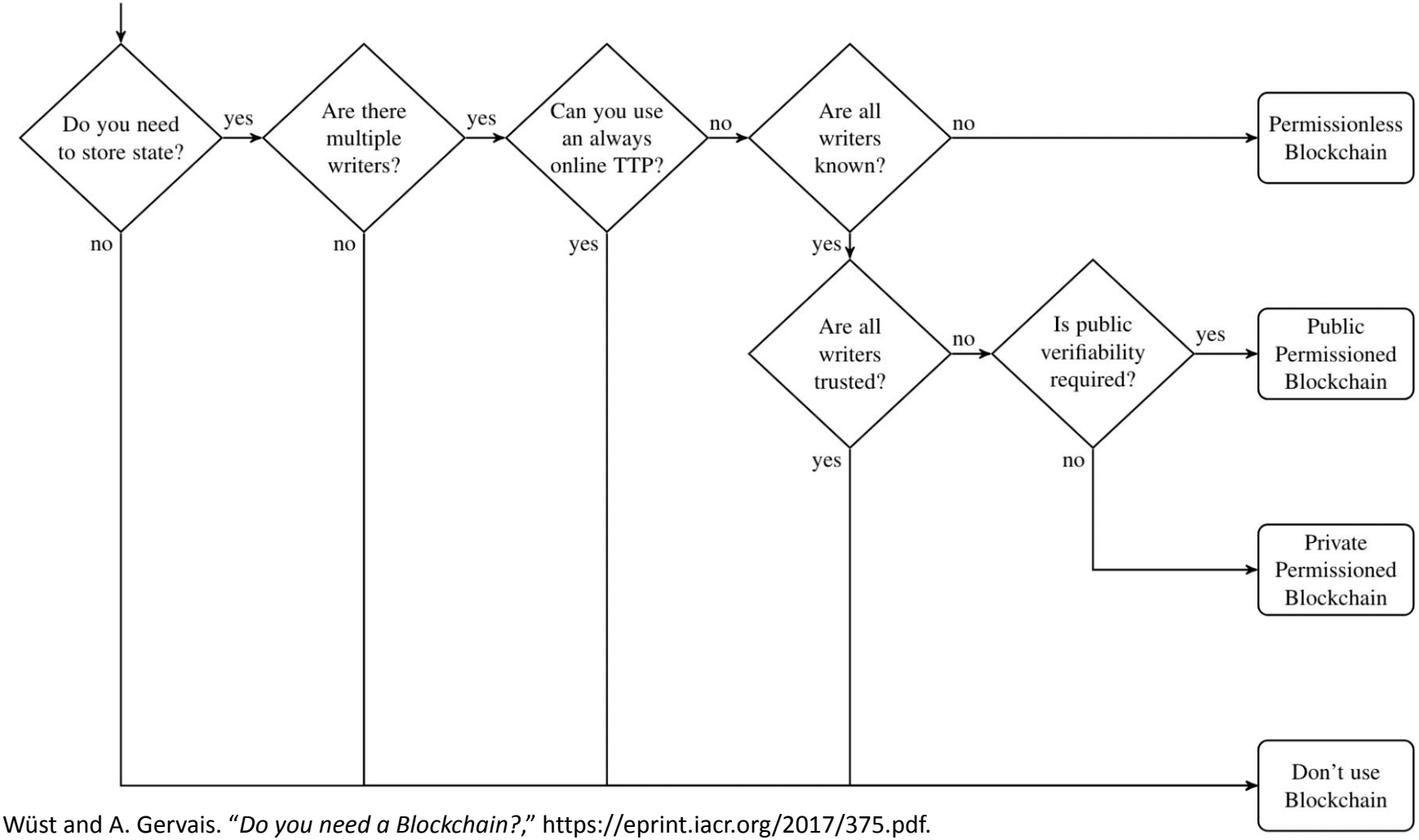
Building blocks

- Mining pools
  - Probability of finding a block alone is very small
  - Unite in Mining pools
  - Payout is done proportional to the work



Need of Blockchains

- Possible decision tree



K. Wüst and A. Gervais. "*Do you need a Blockchain?*," https://eprint.iacr.org/2017/375.pdf.

Need of Blockchains

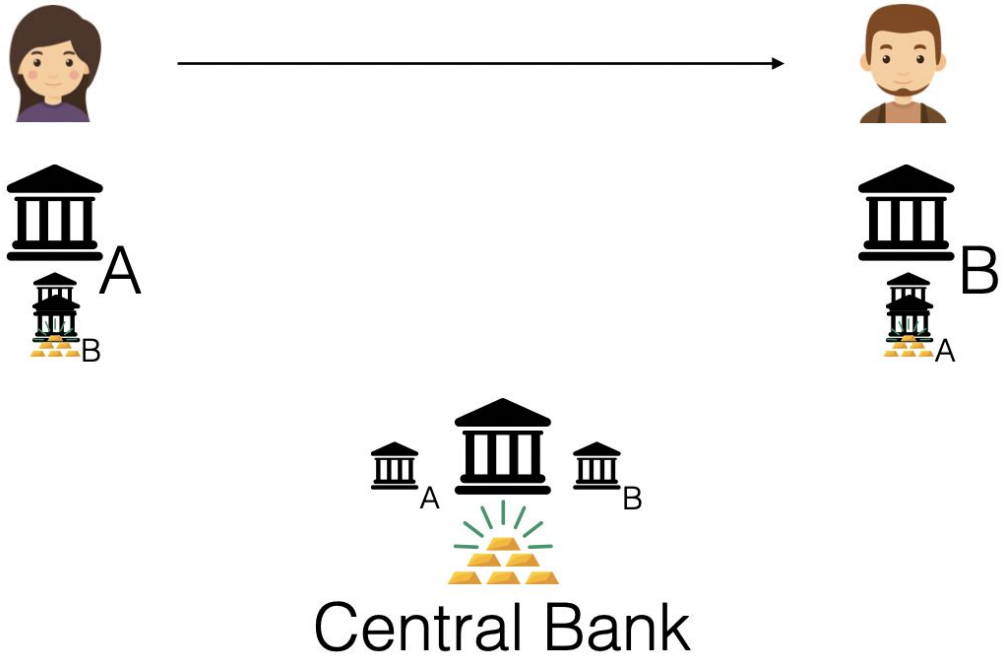
- Typical scenarios
  - International banking

  - Supply chain management

  - Ransomware

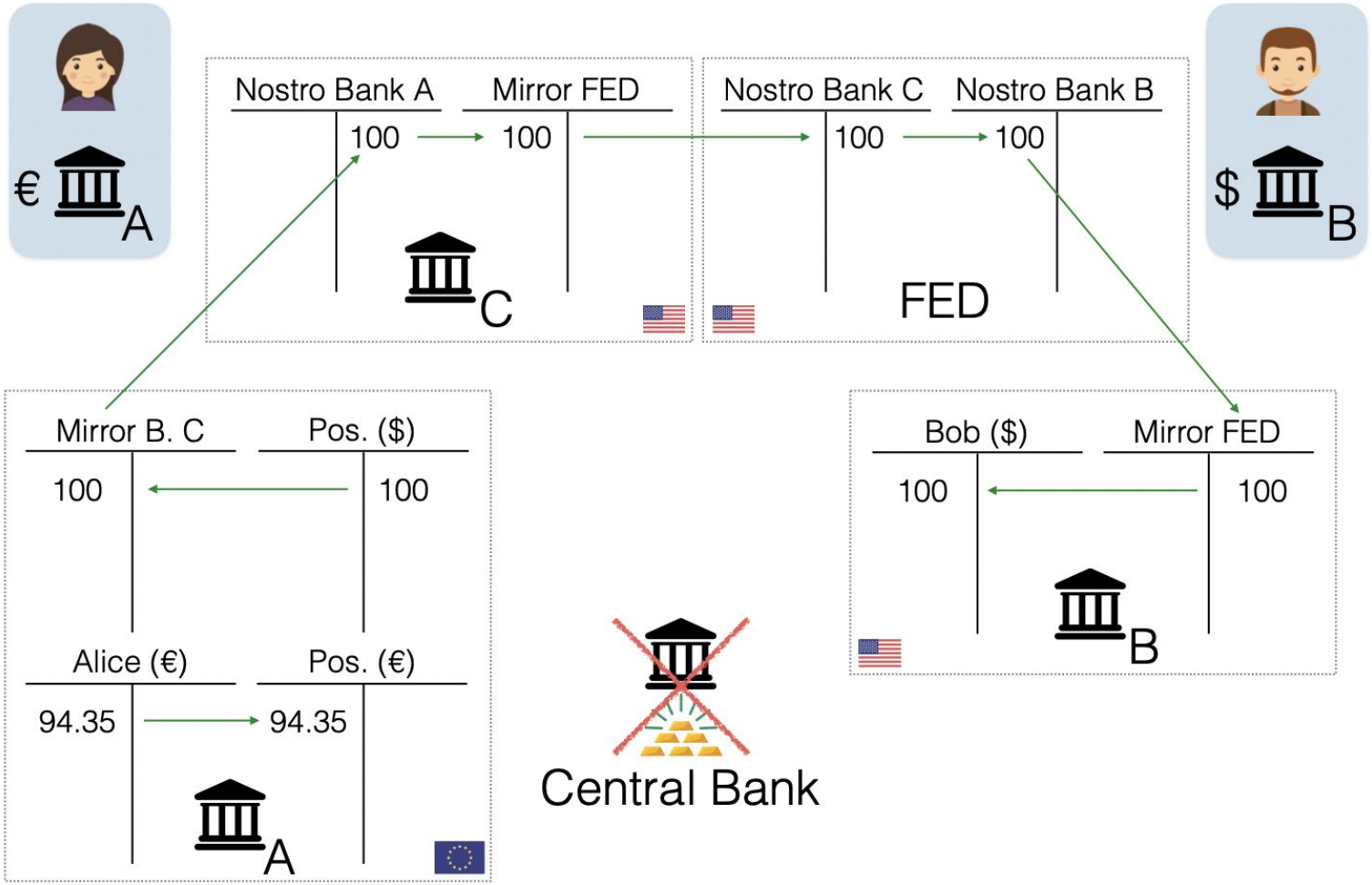
Need of Blockchains

- Bank payments
  - Transfer within a Bank (A —> B)
    - Simple change of books
      - Bank's total balance doesn't change
  - Transfer across Banks



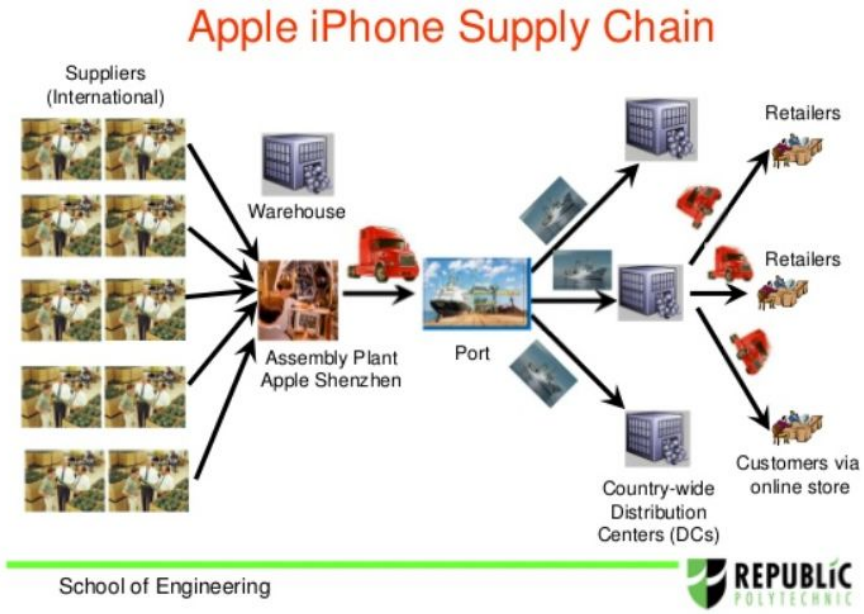
Need of Blockchains

- International payments



Need of Blockchains

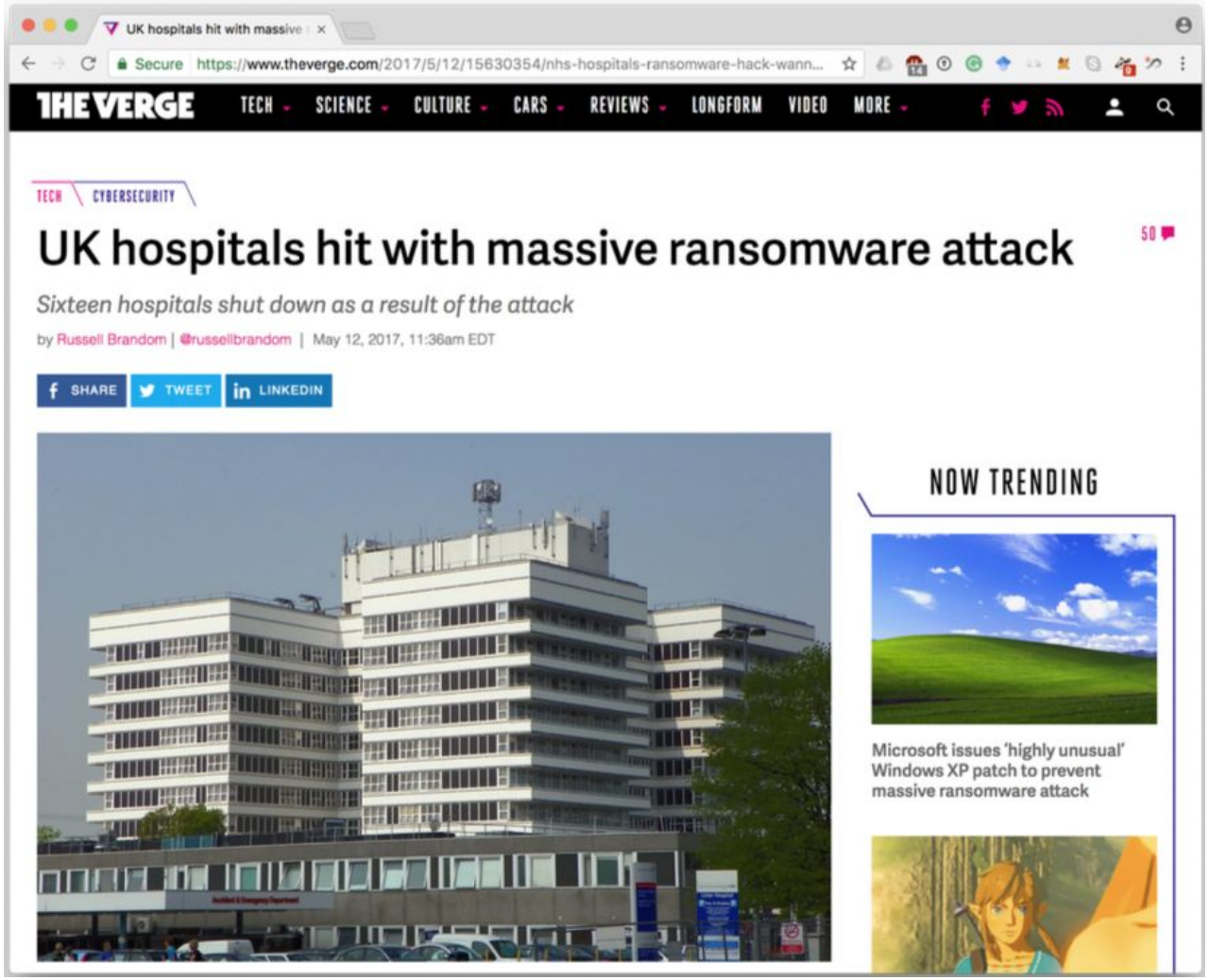
- Supply chain management
  - Producing an iPhone involves many parties (> 500 suppliers)



  - Supply chain
    - Estimate future demand based on past and current demand
      - What are the fundamental challenges of SCM & blockchain?

Need of Blockchains

- Ransomware



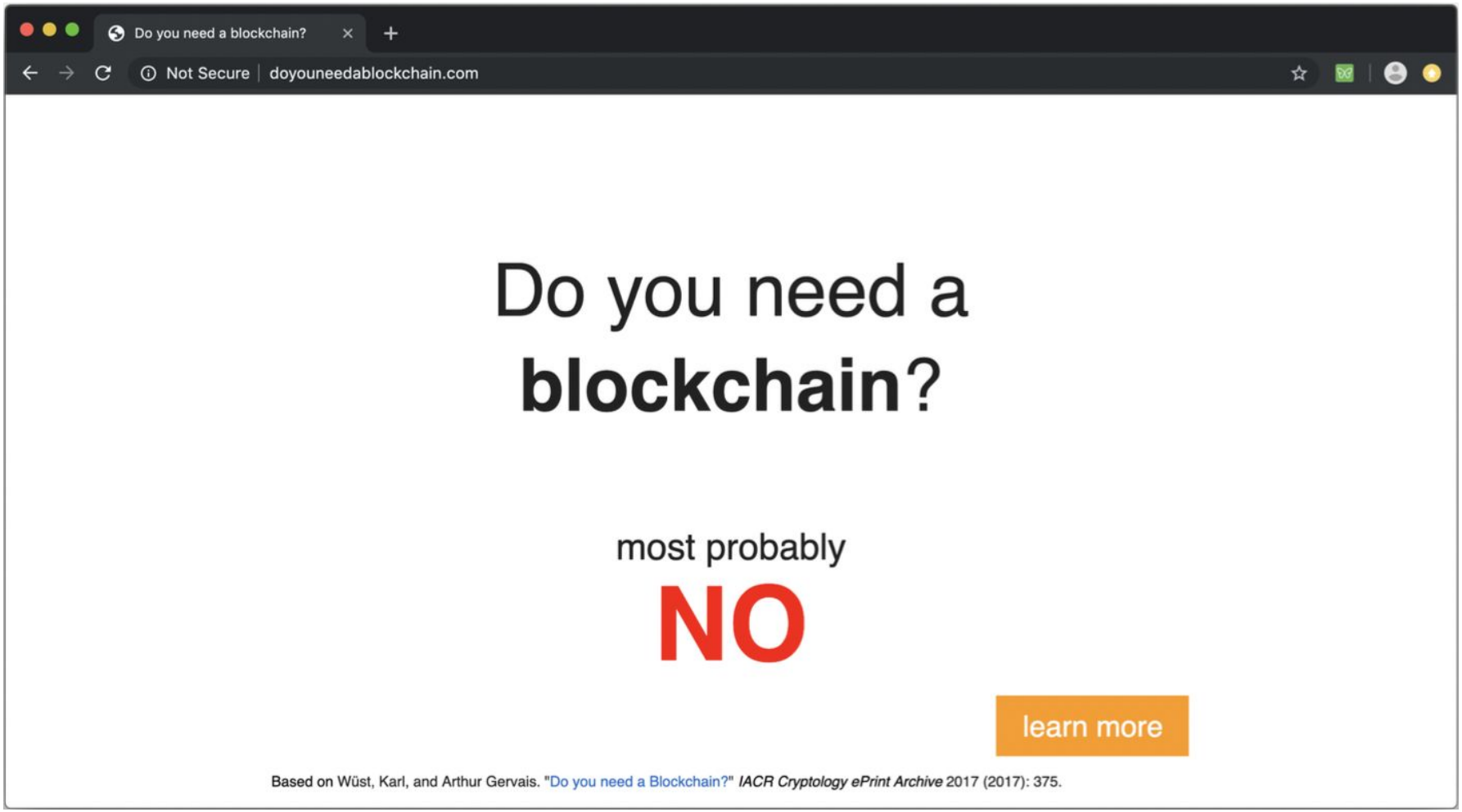
Need of Blockchains

- The future of Ransomware
  - Ransomware requires
    - A key to encrypt the victim's data
    - A payment mechanism

```
if(victim paid) {
  Reveal key;
  Victim can decrypt data;
} else {
  Don't reveal the key;
}
```

Done manually!  **Autonomous Randomware?**

Need of Blockchains



Scaling Blockchains

- Cryptocurrencies' transaction speeds compared to VISA & Paypal

Scaling Blockchains

- Cryptocurrencies do not scale

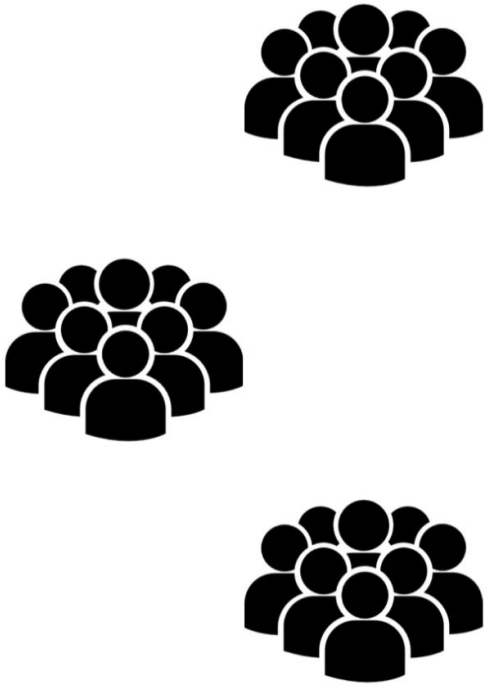
Size of tx in KB

~ 10 tps

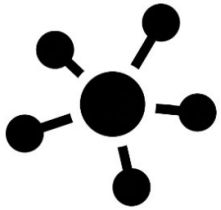
Complexity of tx in “Gas”

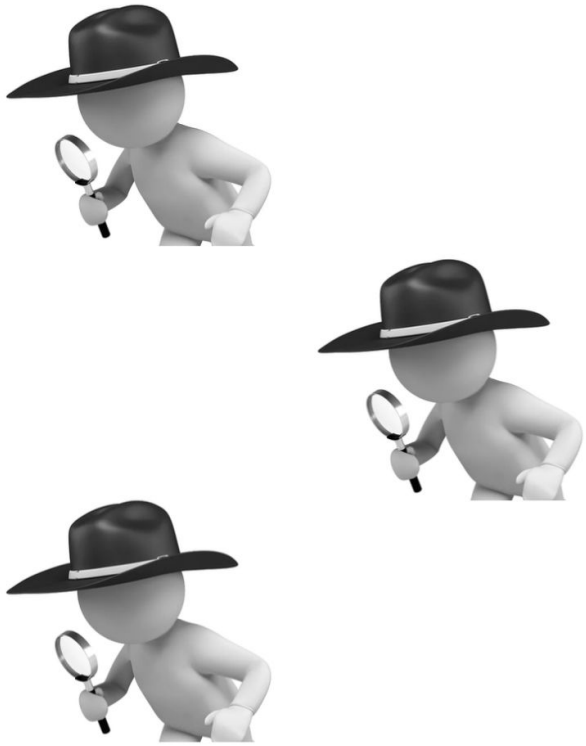
Scaling Blockchains

- Why doesn't it scale?



Many users

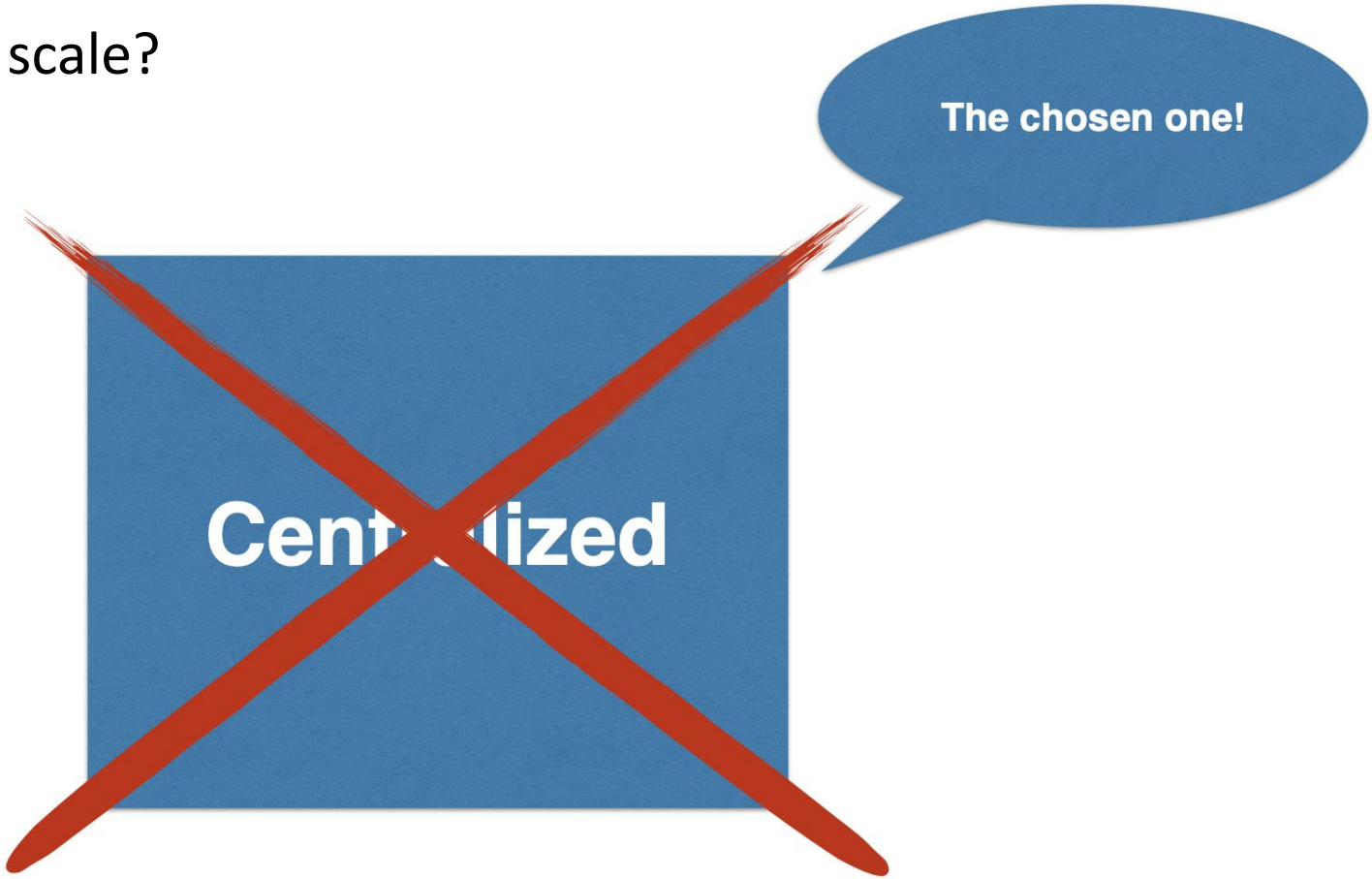




Many validators

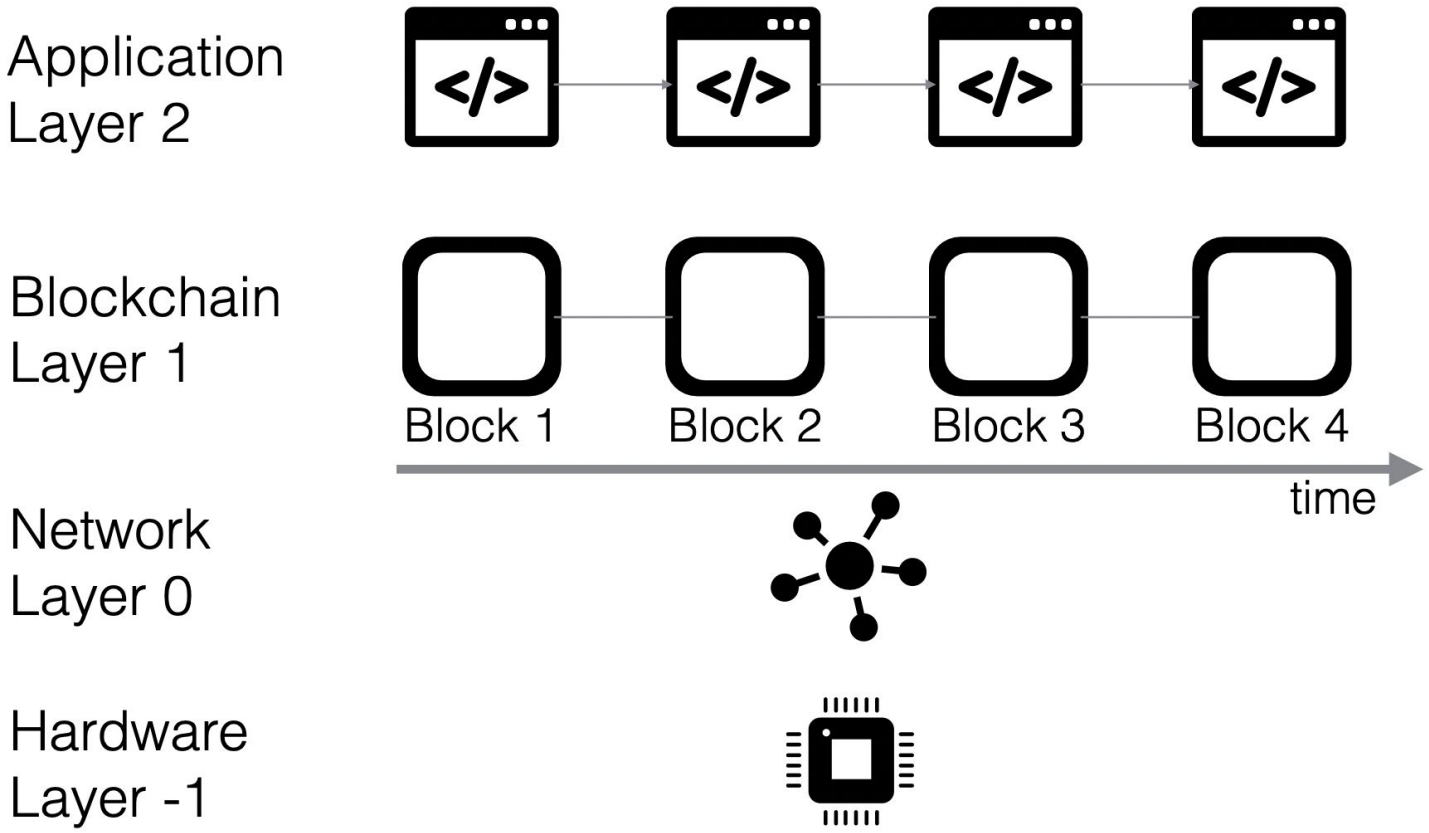
Scaling Blockchains

- How can we scale?



Scaling Blockchains

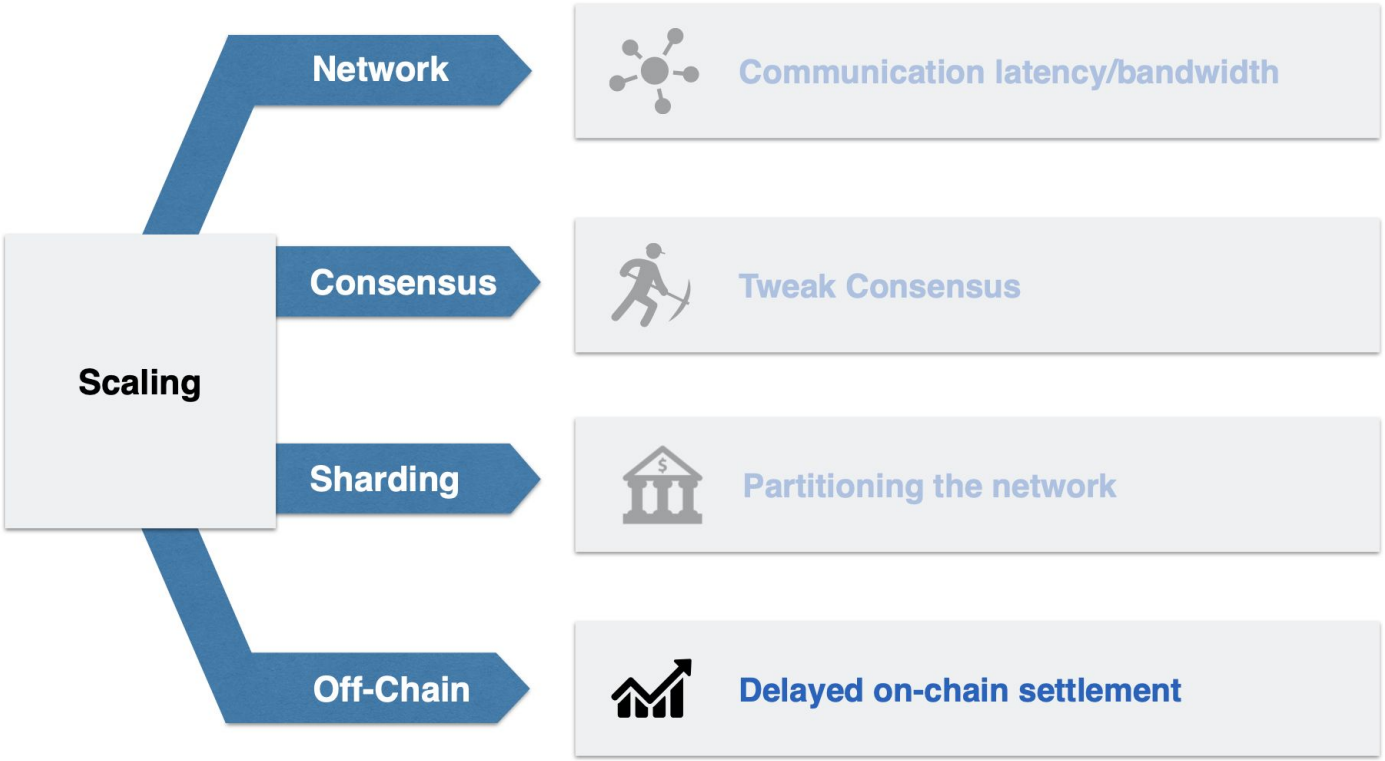
- Blockchain layers



Gudgeon et al. *"Sok: Layer-two Blockchain Protocols,"* Financial Cryptography, 2020.

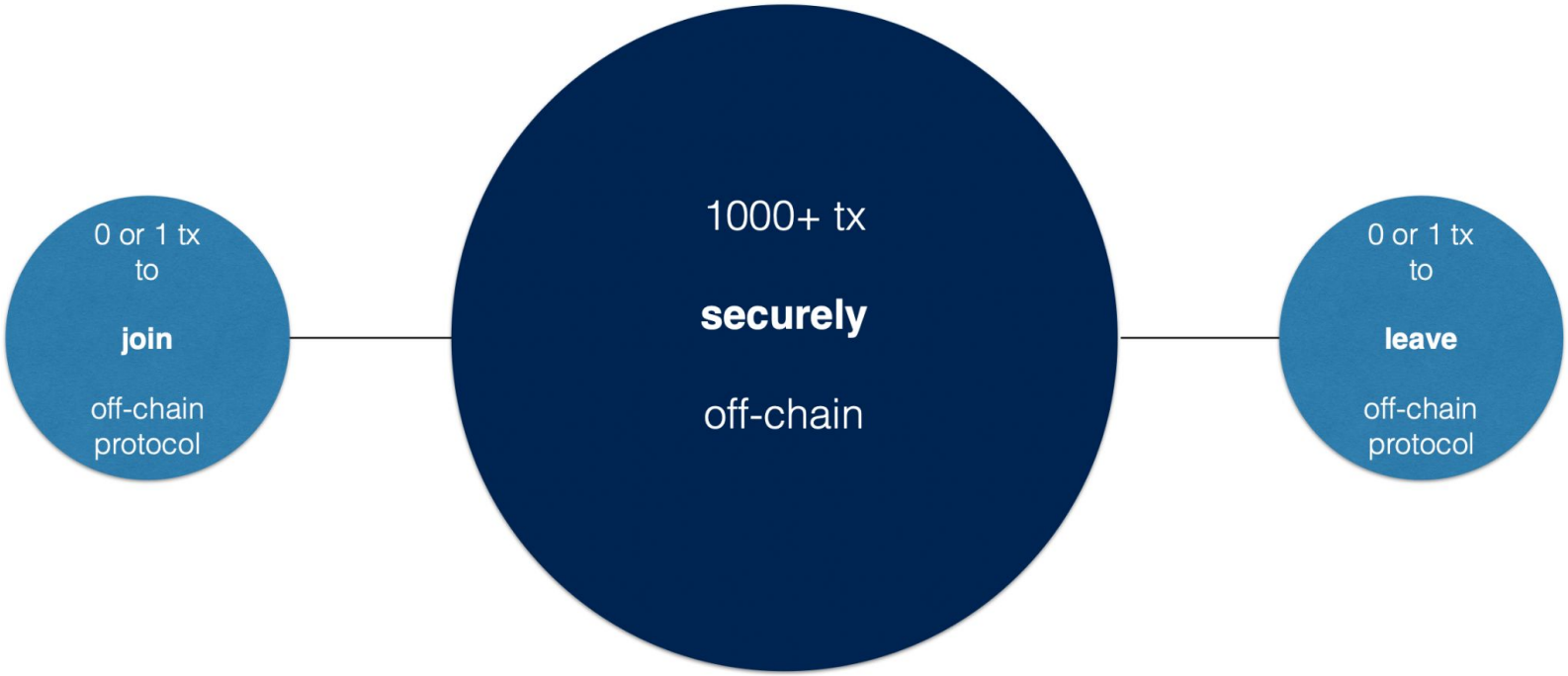
Scaling Blockchains

- How can we scale?



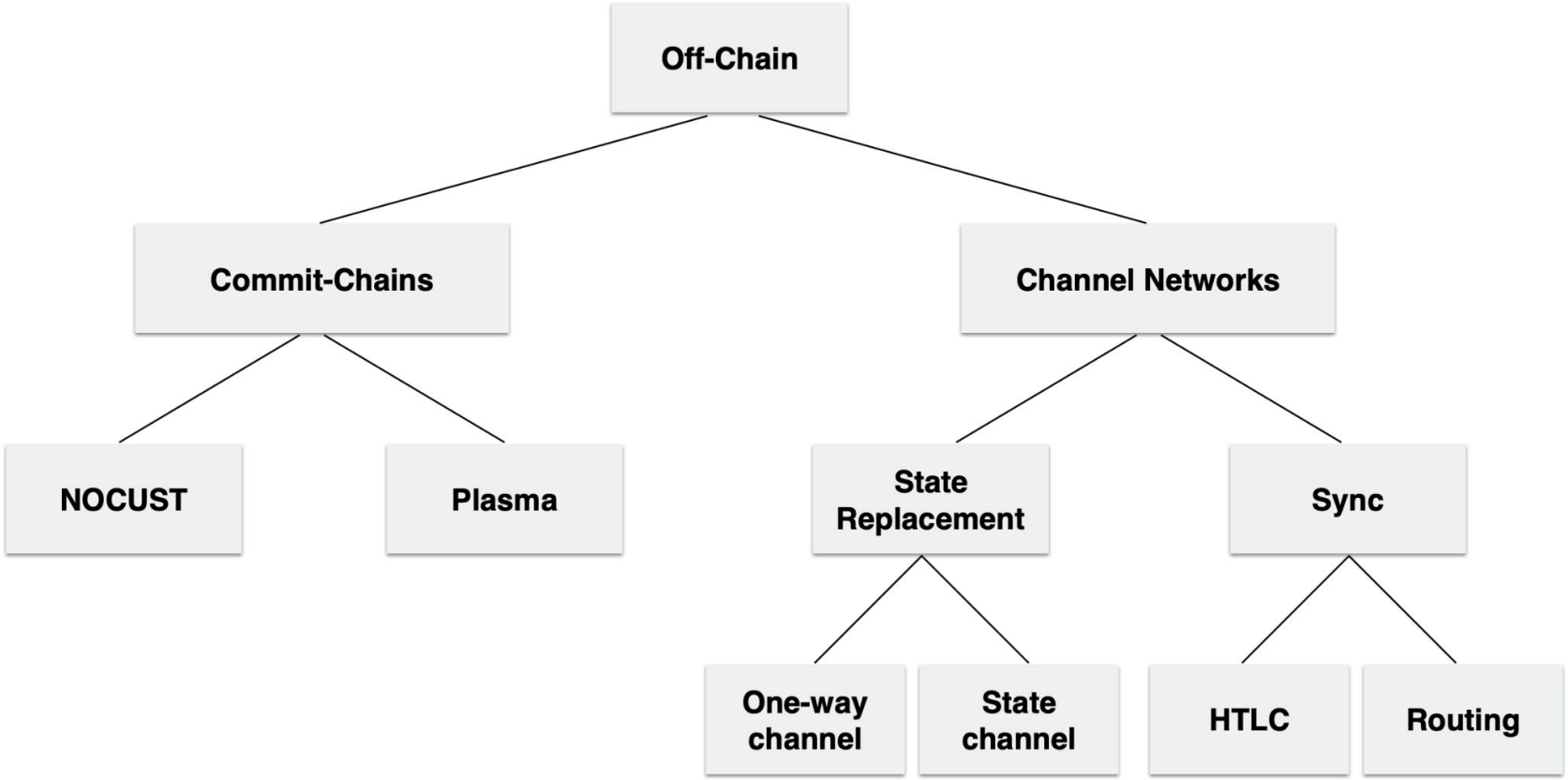
Scaling Blockchains

- Why is off-chain exciting?
  - No consensus latency or mining fees
    - While still achieving non-custodial security
  - Backward compatibility!



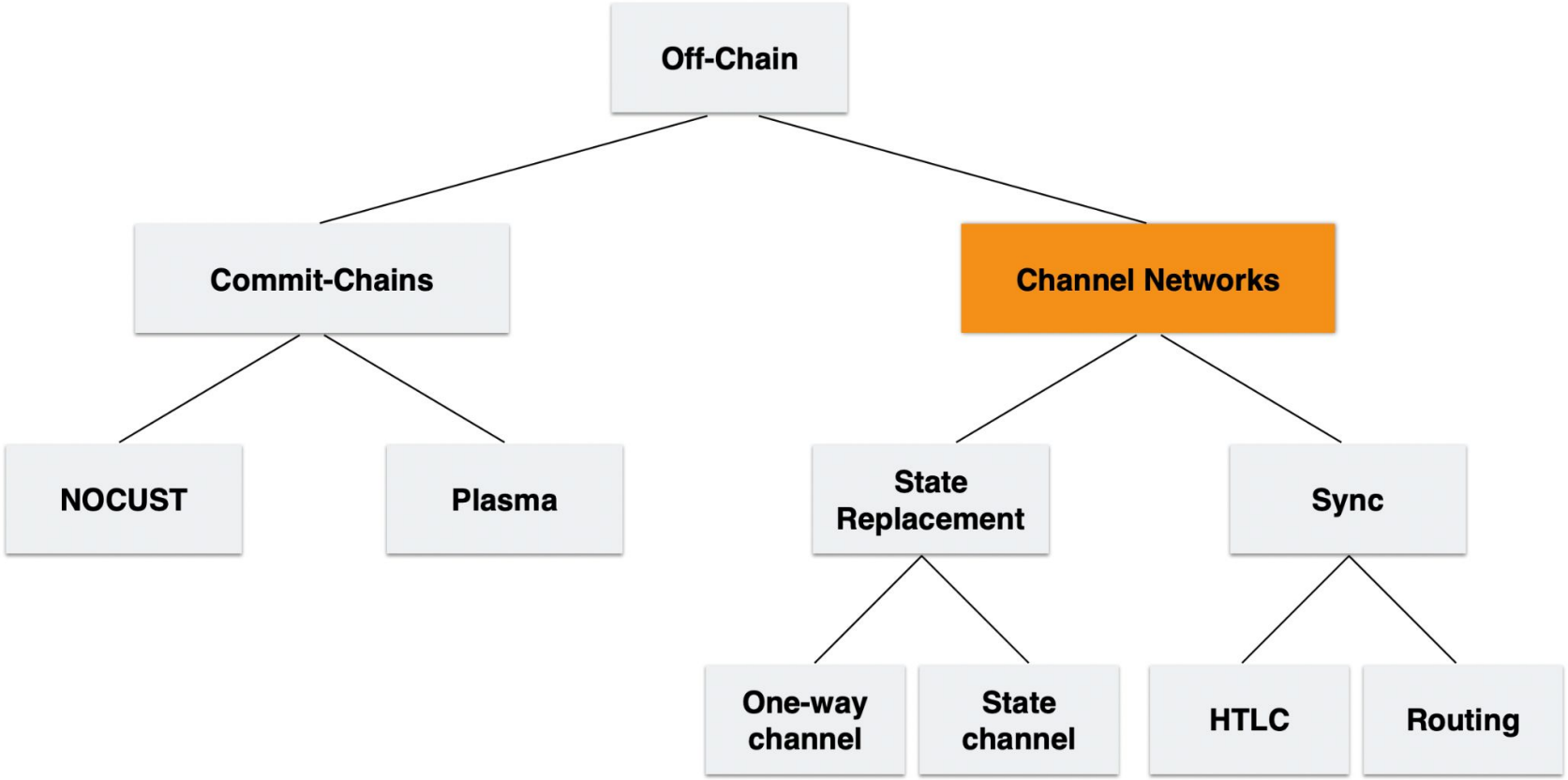
Scaling Blockchains

- Which off-chain solution?



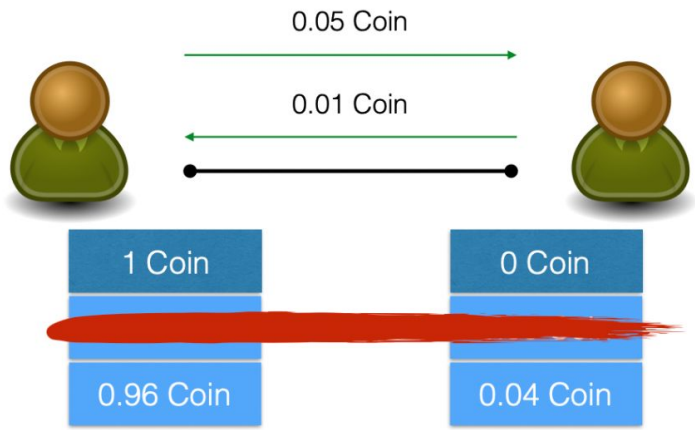
Scaling Blockchains

- Which off-chain solution?



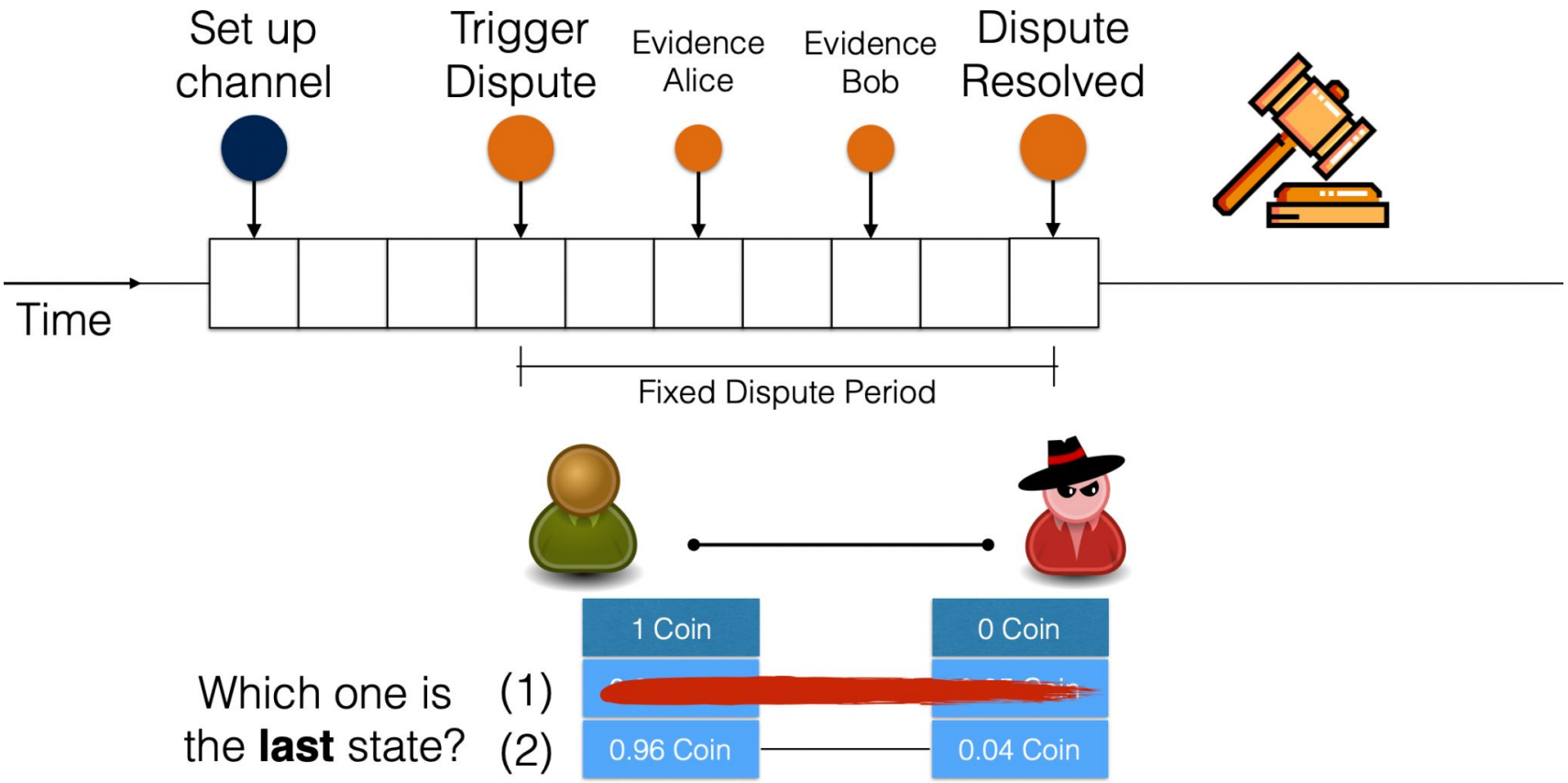
Scaling Blockchains

- Payment channel



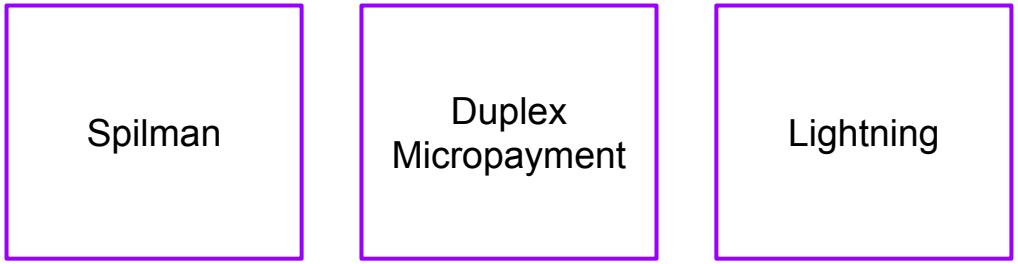
Scaling Blockchains

- Payment channel
  - Dispute

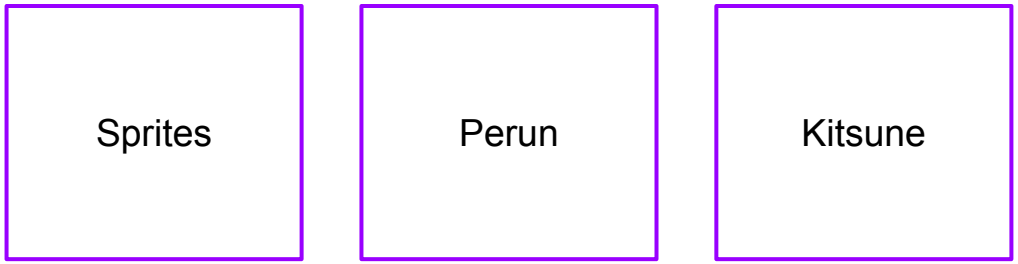


Scaling Blockchains

- Payment channel

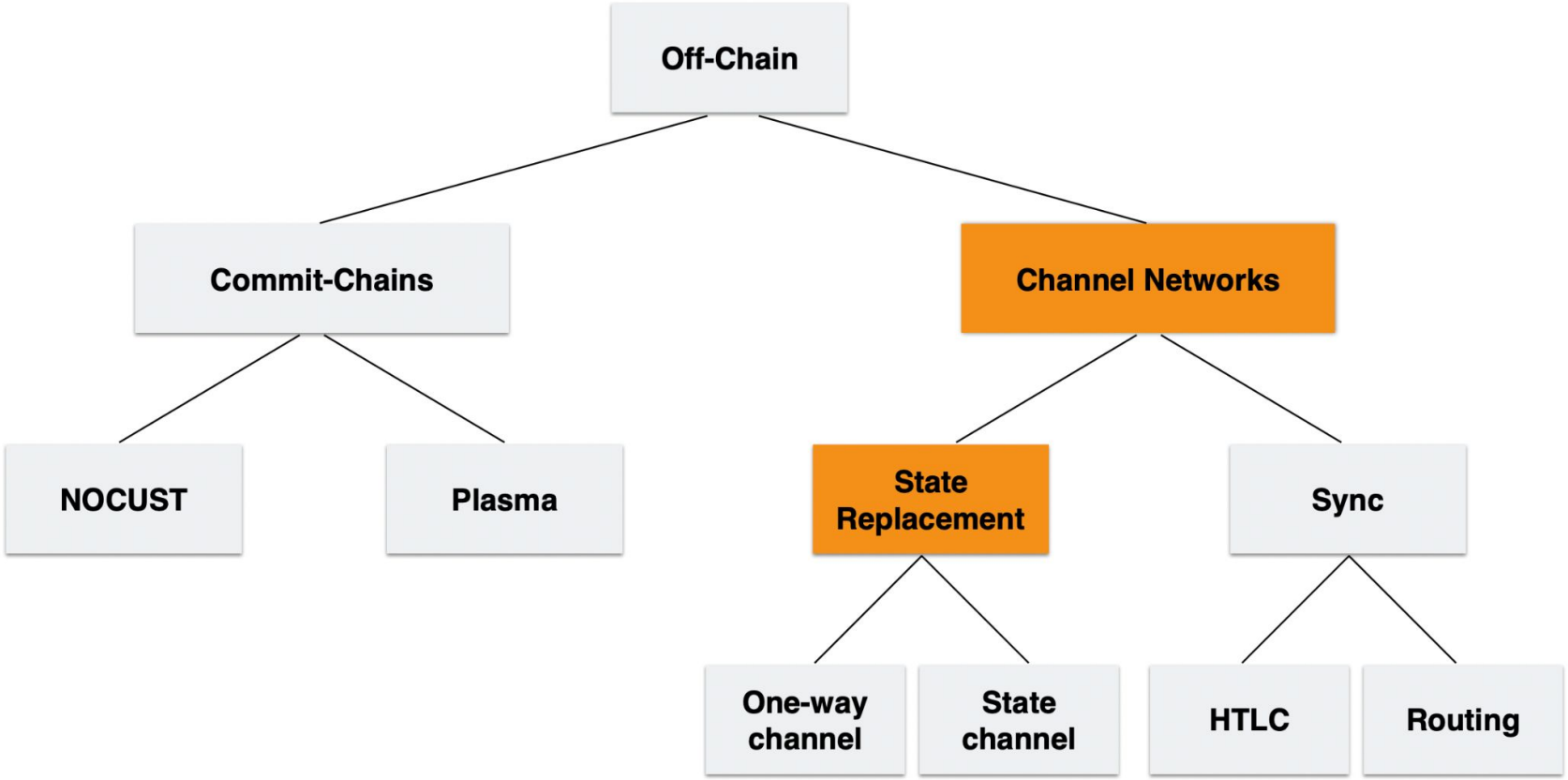


- State channel (game, voting, auctions, etc.)



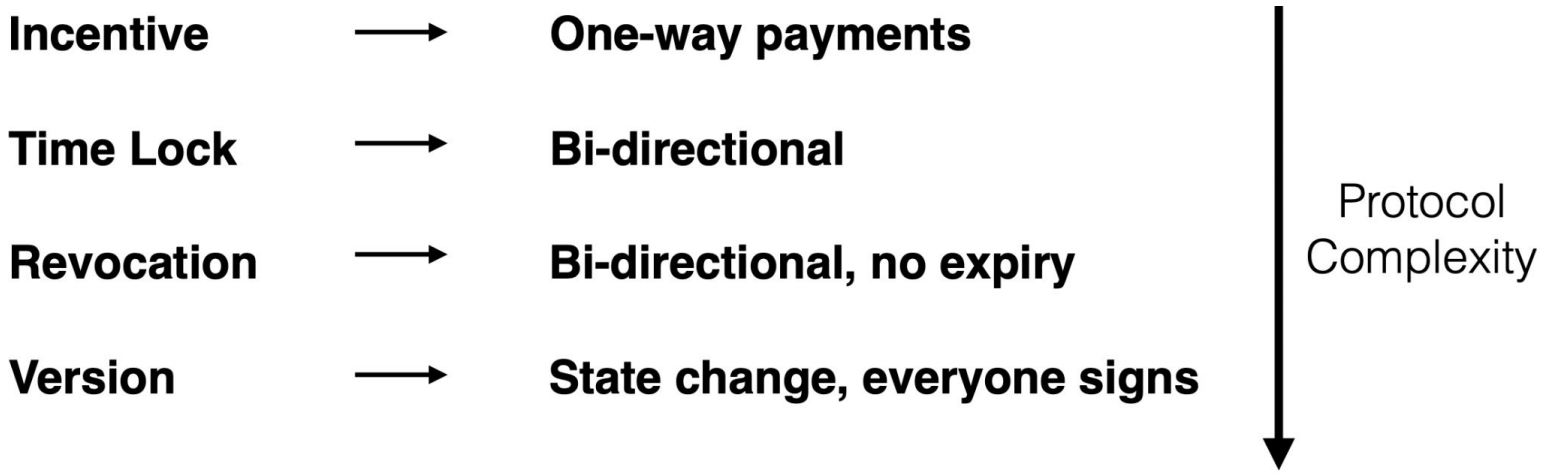
Scaling Blockchains

- Which off-chain solution?



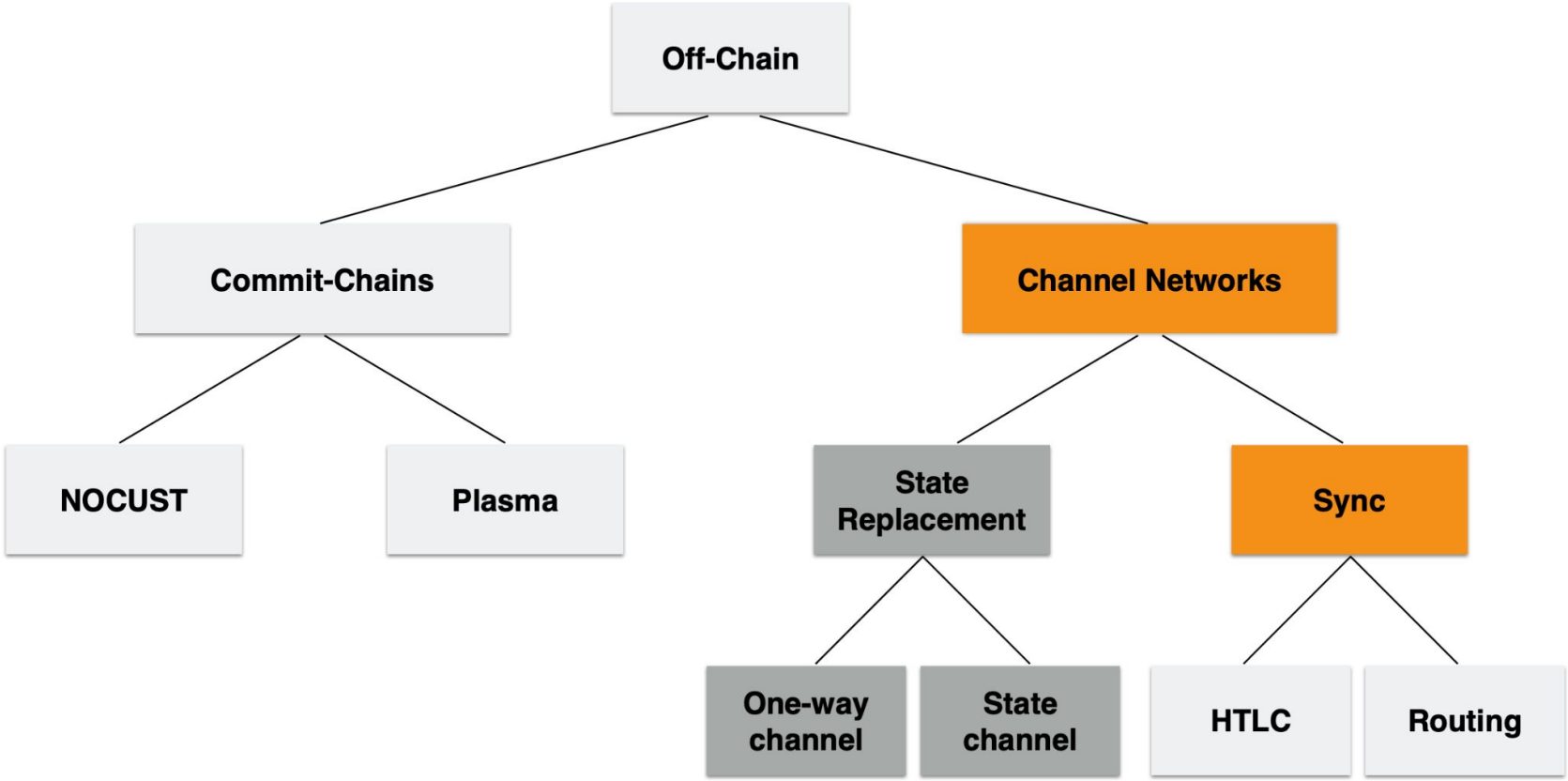
Scaling Blockchains

- State replacement techniques
  - Replace by



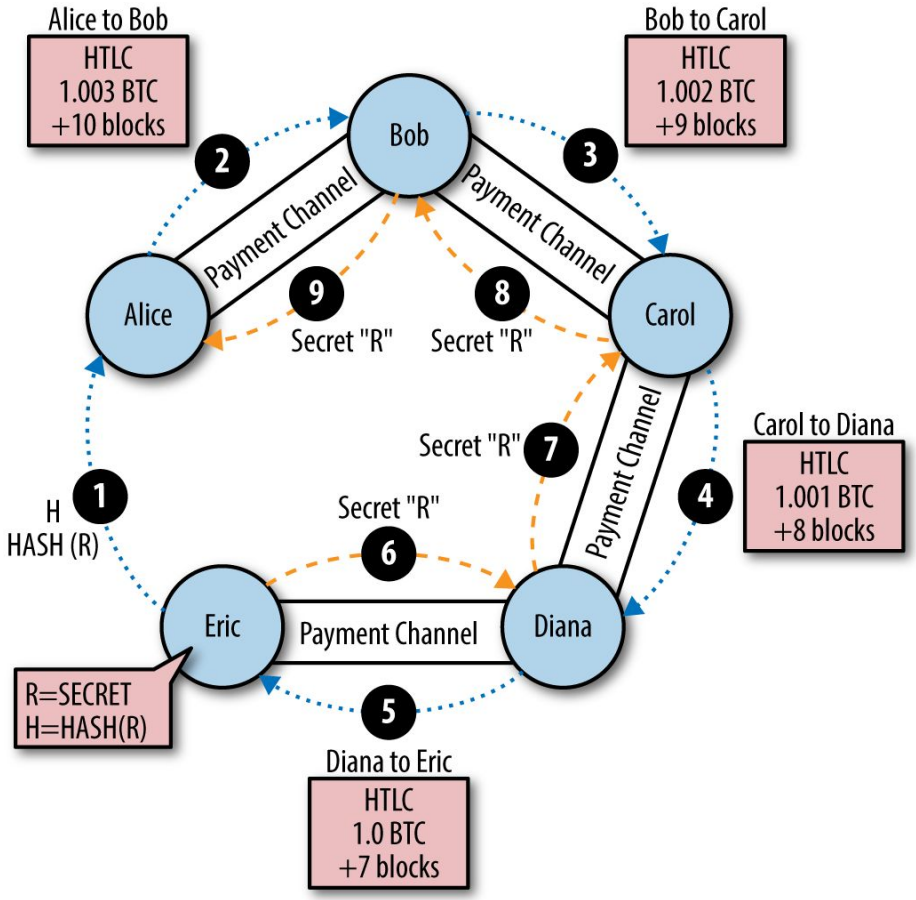
Scaling Blockchains

- Which off-chain solution?



Scaling Blockchains

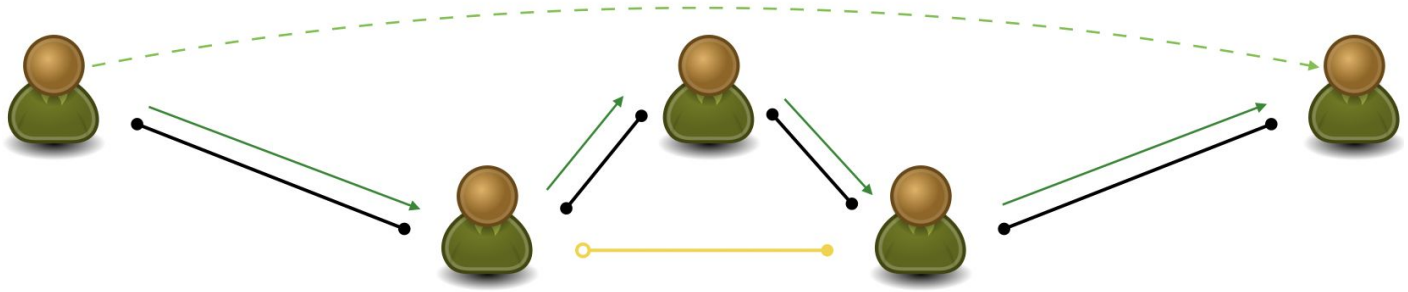
- Payment channel
  - Hash Time Locked Contracts (HTLC)



https://medium.com/softblocks/lightning-network-in-depth-part-2-htlc-and-payment-routing-db46aea445a8

Scaling Blockchains

- Payment channel
  - Routing



HTLC + Route Finding

Scaling Blockchains

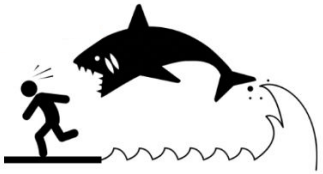
- The good, bad, and ugly of payment channels

Collateral for each hop

No direct connection needed

Decentralized, limited censorship

On-chain channel establishment

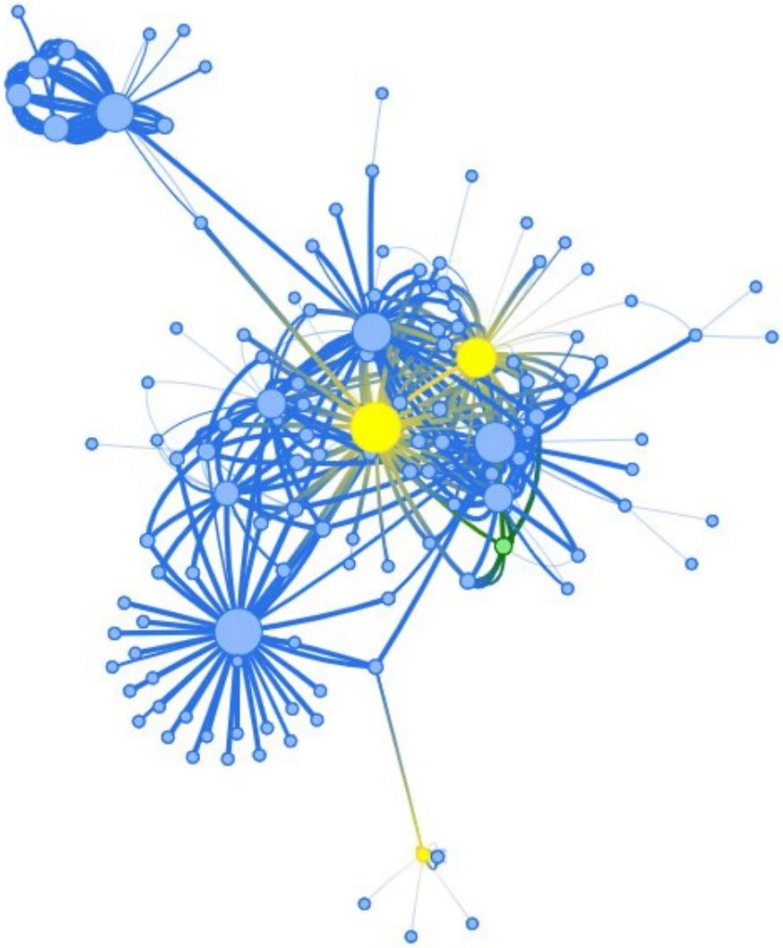


Wormhole attacks

Optimistically fast and cheap

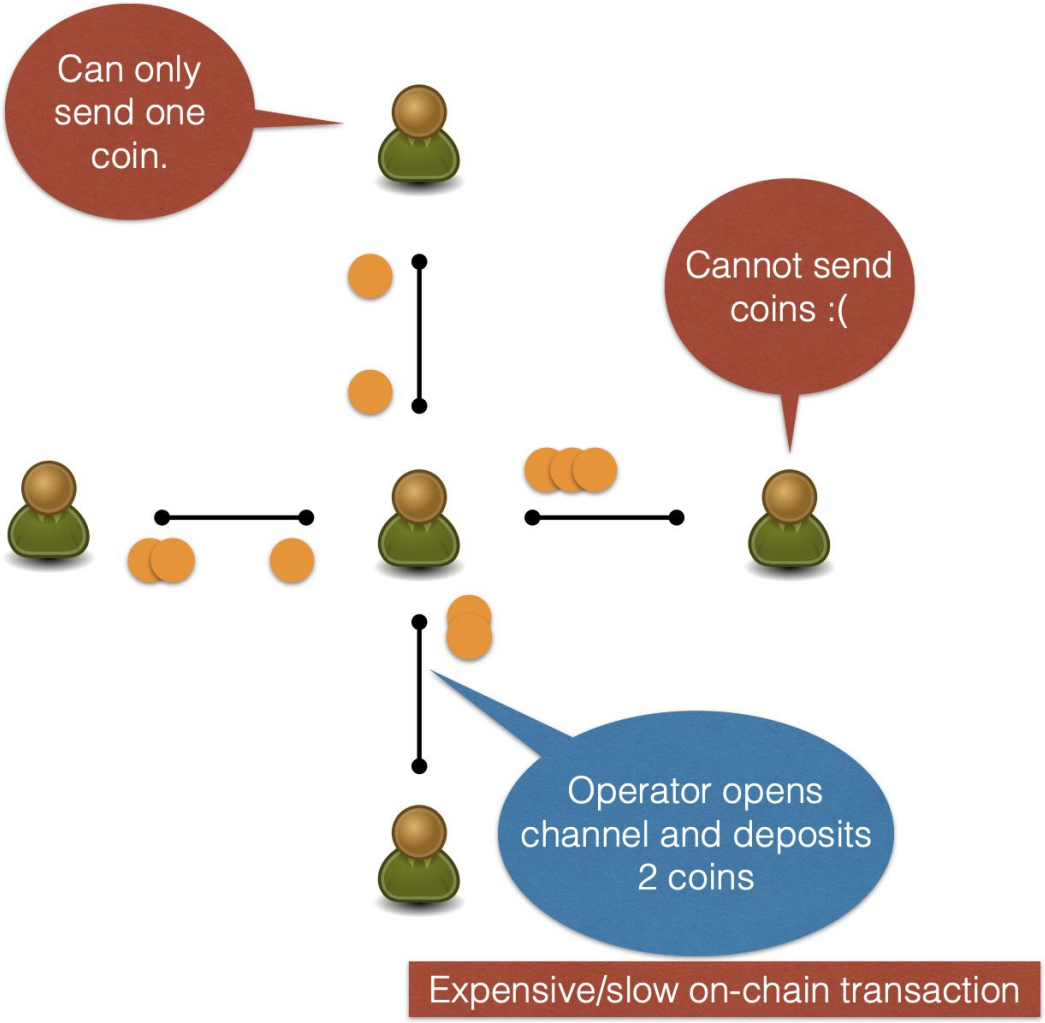
Scaling Blockchains

- Star topologies in theory & practice



Scaling Blockchains

- Payment channel hub

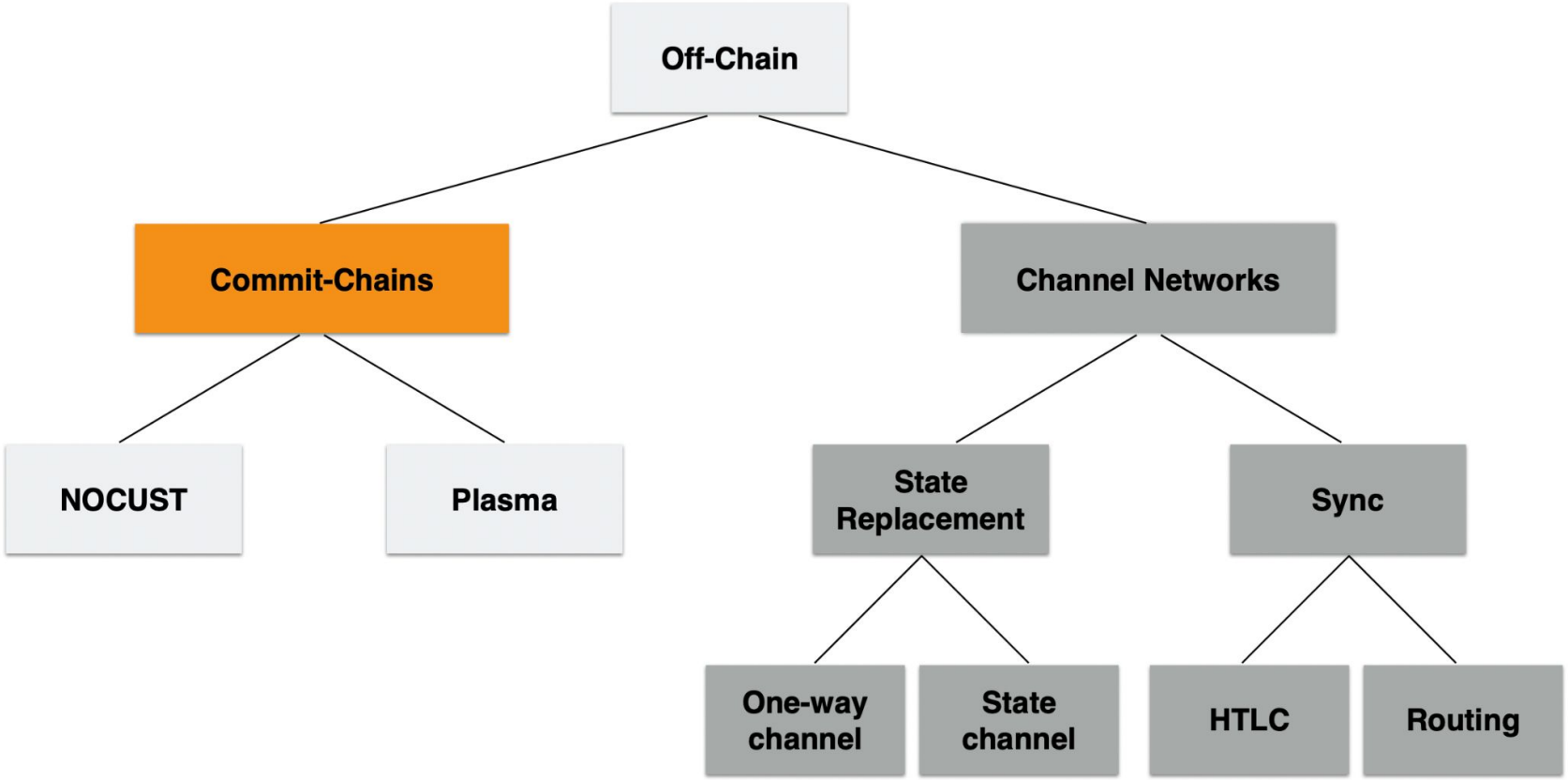


Scaling Blockchains

- Payment channel hub
  - Are great for instant finality and no trust
  - But, expensive to run
  - Can we do better with eventual finality?

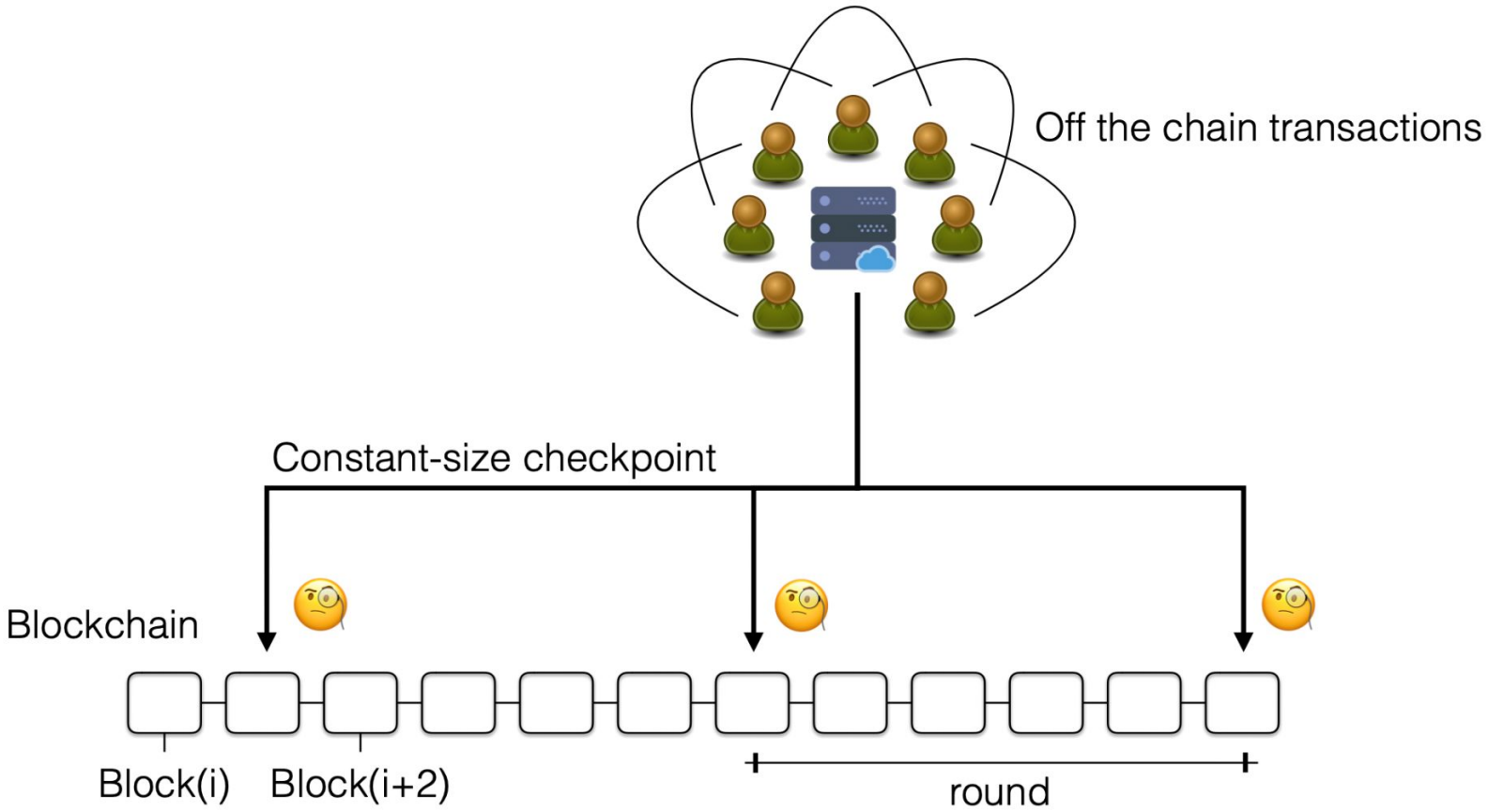
Scaling Blockchains

- Which off-chain solution?



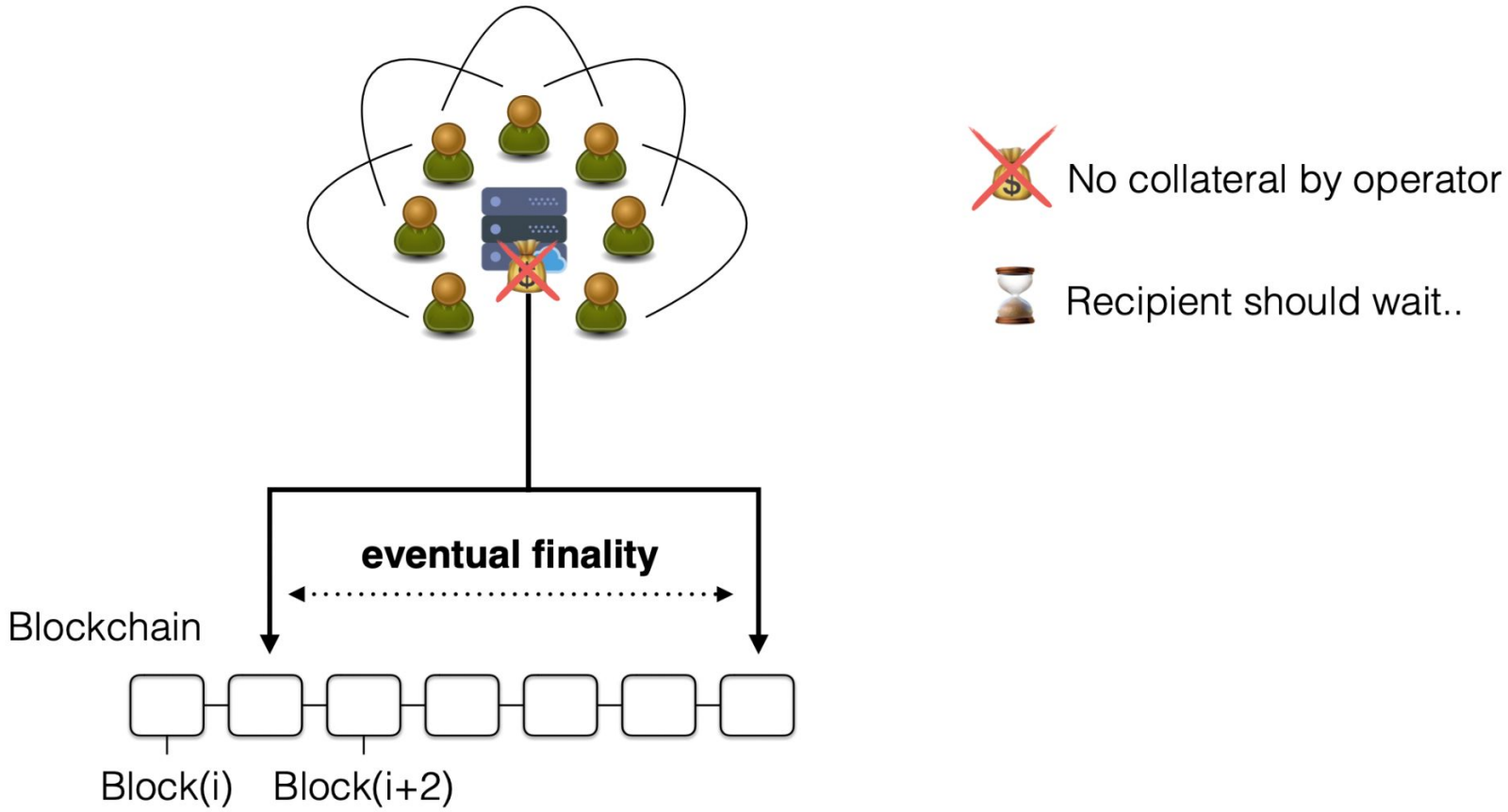
Scaling Blockchains

- Commit-Chains



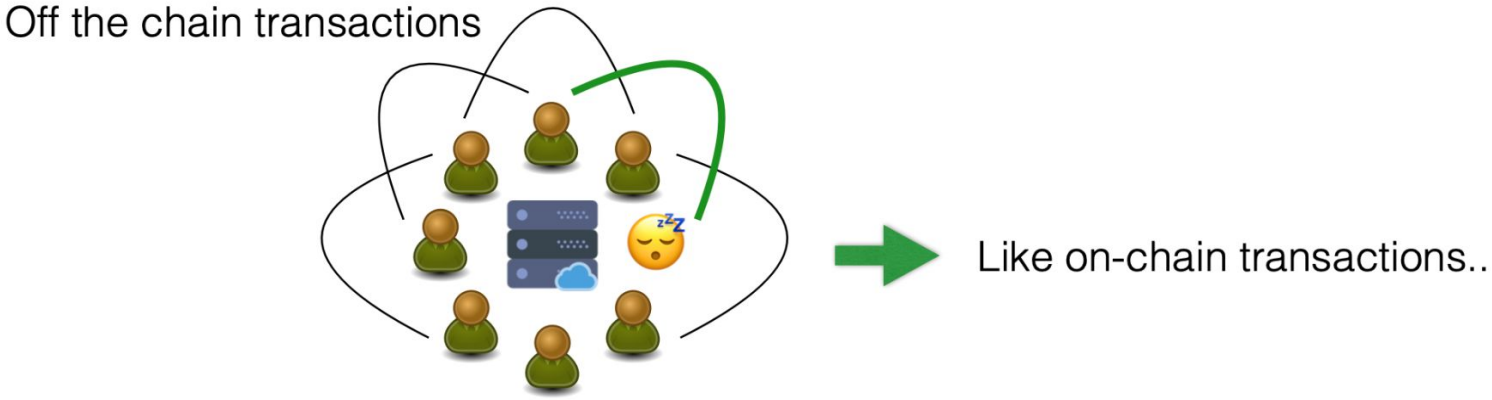
Scaling Blockchains

- Without collateral —> Eventual finality



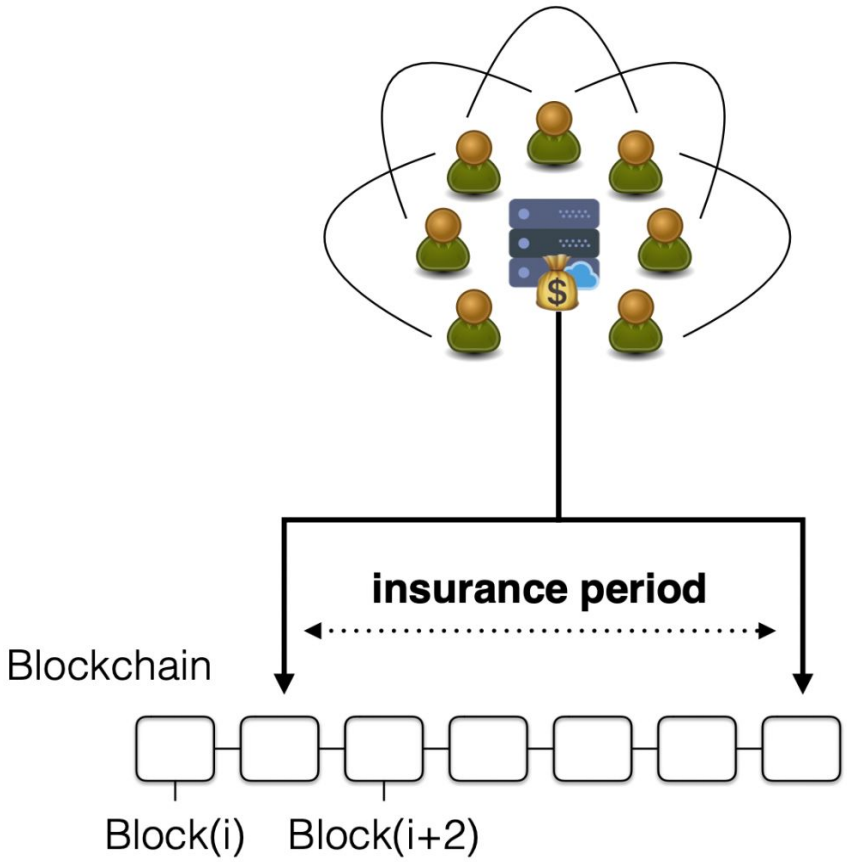
Scaling Blockchains

- Receive TX while offline



Scaling Blockchains

- With collateral —> Instant finality



Insurance pool

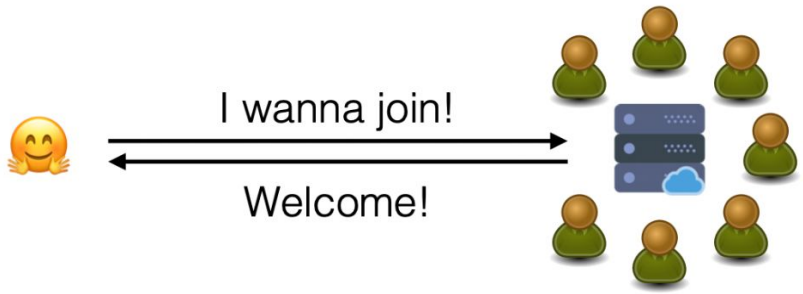
Collateral re-usable each round

TX can be accepted instantly

Collateral allocated with 1 on-chain TX

Scaling Blockchains

- Join without on-chain Transaction

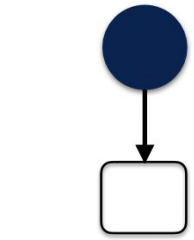


- Instant
- CO_2 friendly (Zero gas costs)

Scaling Blockchains

NOCUST

Off-chain ledger state:
user balances



checkpoint

Plasma (Cash)

Off-chain ledger state:
serial number coins

- Fungible payments
- Slower delayed finality
- Lightweight clients
- Instant finality support
- Atomic Off-chain Swap (TEX)

- Non-Fungible payments
- Rapid delayed finality
- Large amounts of history data
- No instant finality support
- No known feasibility for swaps

Scaling Blockchains

- A centralized operator
  - Untrusted and non-custodial!

Can

Censor off-chain tx

Cannot

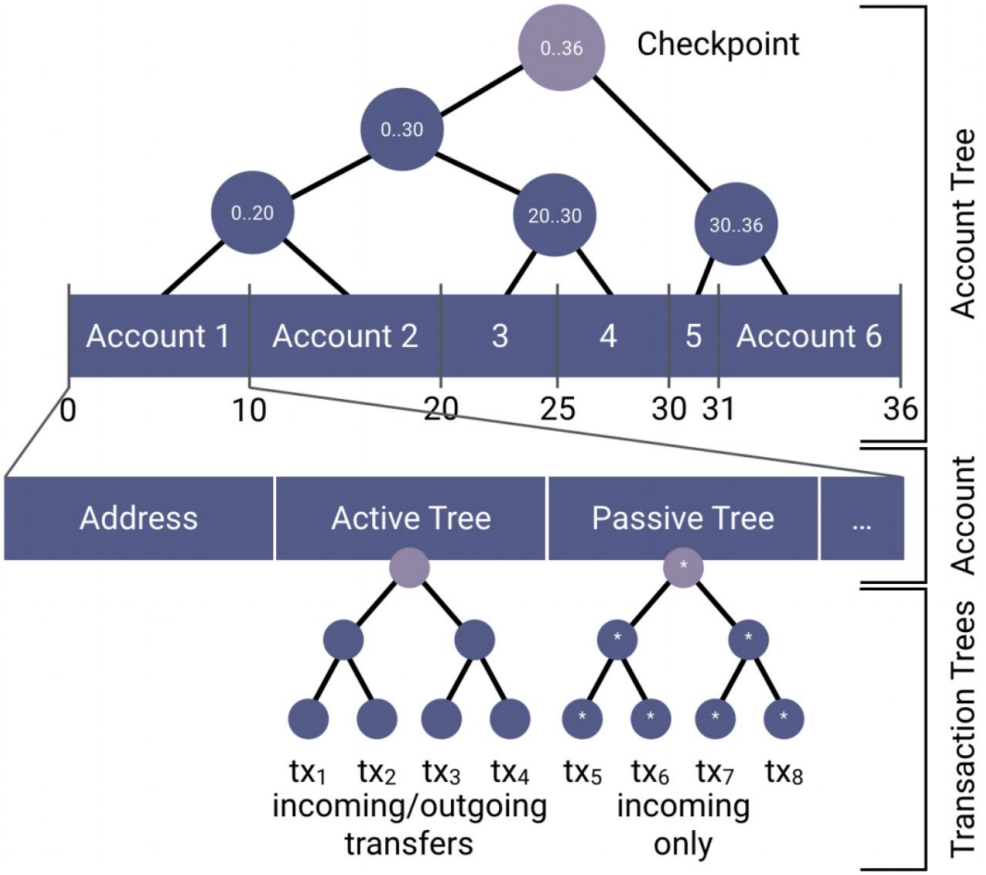
Steal coins

Double spend coins

Mint new coins

Scaling Blockchains

- NOCUST key innovation
  - Merkle interval tree

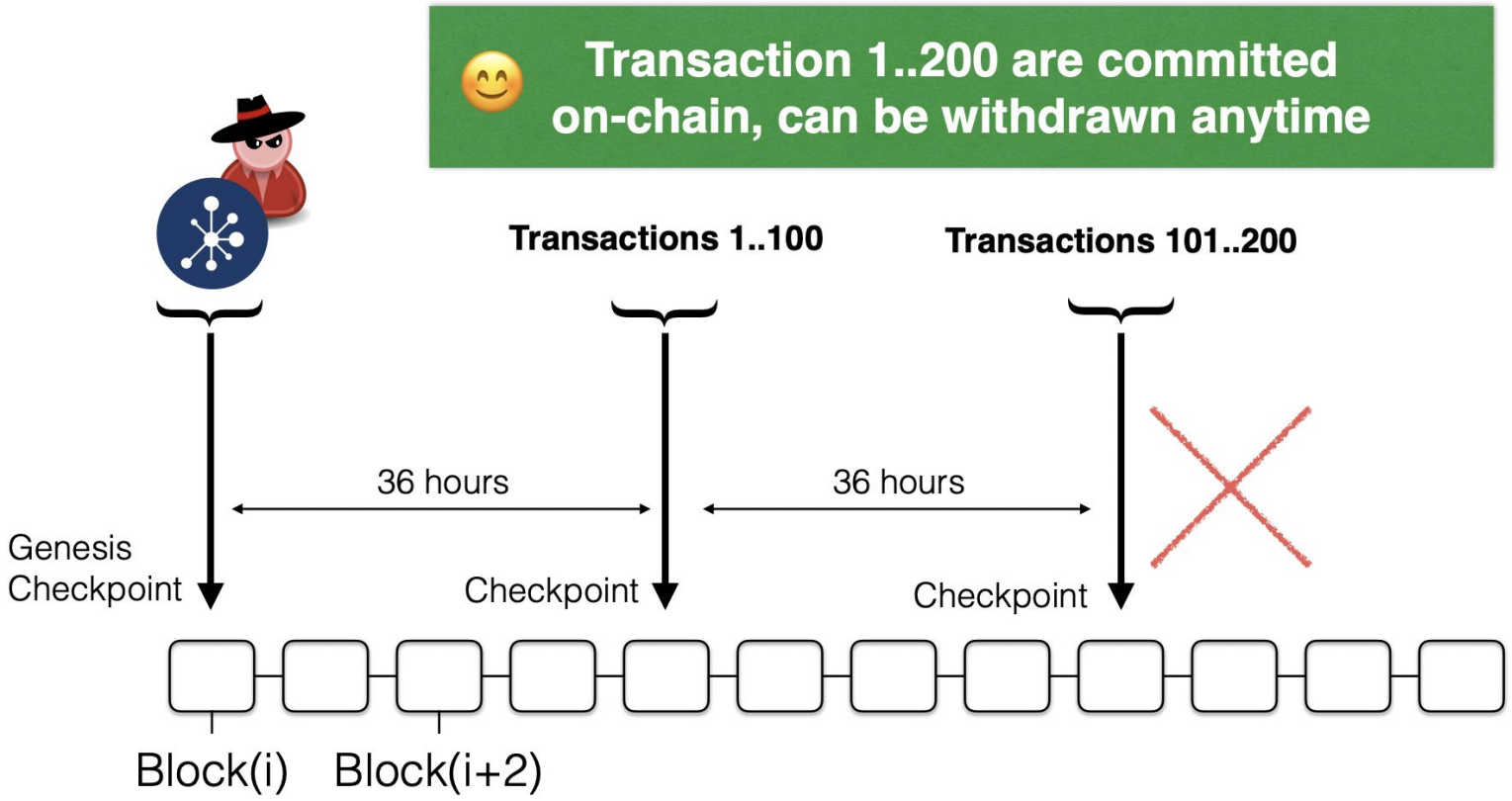


Scaling Blockchains

- NOCUST security properties

Scaling Blockchains

- NOCUST operator disappears
  - Just after checkpoint



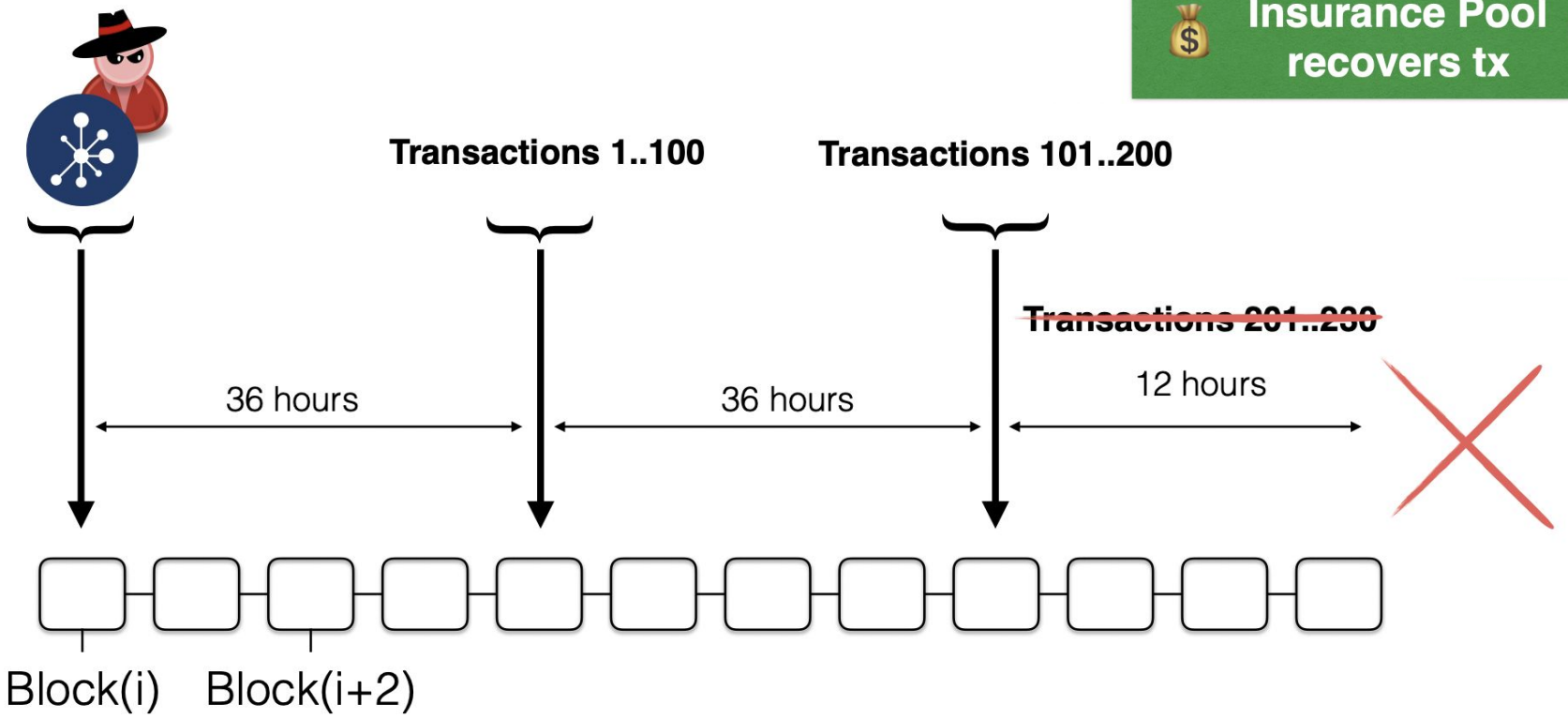
Scaling Blockchains

- NOCUST server disappears
  - After a few transactions

Transaction 1..200 are safe

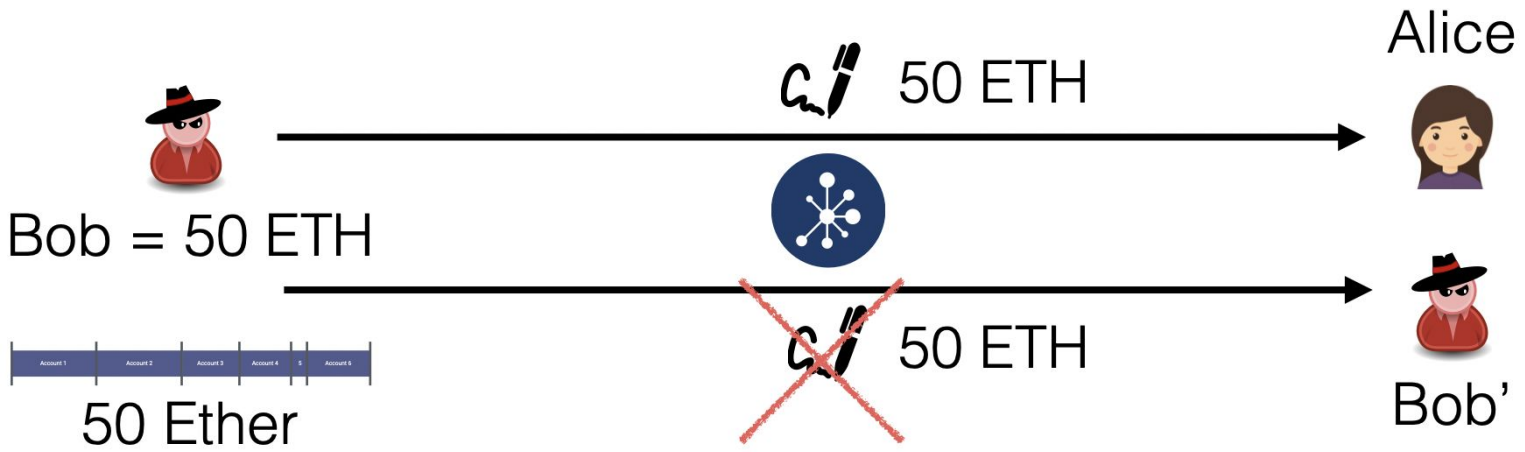
Transaction 201..230 are not committed on chain!

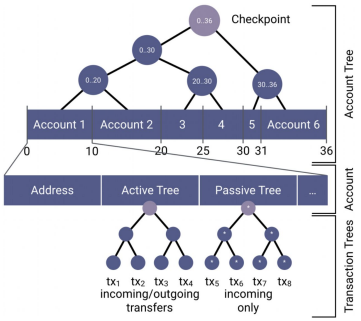
Insurance Pool recovers tx



Scaling Blockchains

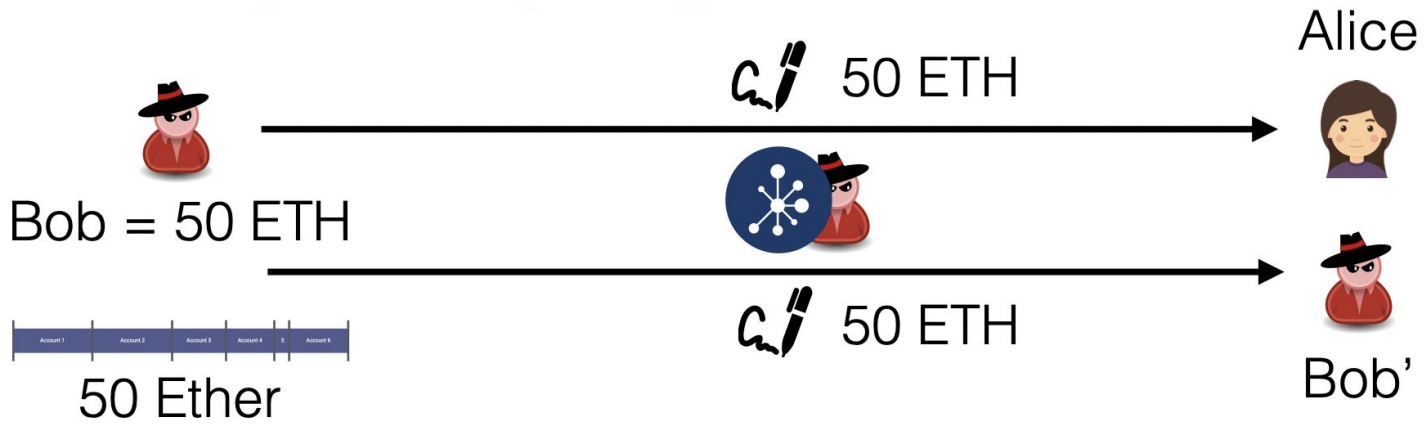
- Users to attempt double-spending





Scaling Blockchains

- 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 !

Scaling Blockchains

- The good, bad, and ugly of commit-chains

Recipient can be offline to receive a transaction

Direct connection needed

Centralized, can censor transactions

No collateral for delayed finality

Instant and free onboarding

Data availability requirements

Scaling Blockchains

- Real-world systems in production