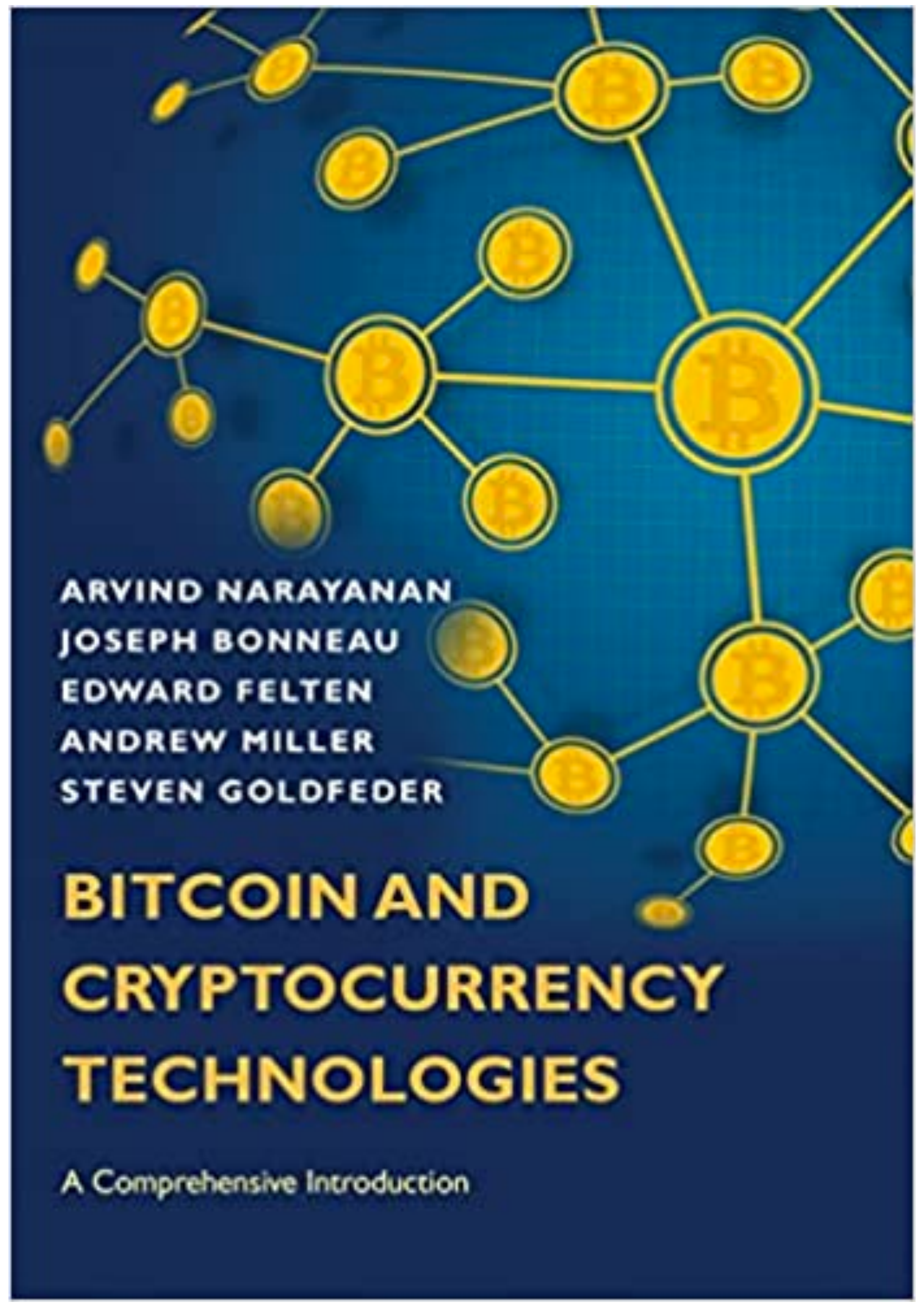
Distributed Systems 2023-2024:
Decentralized systems and Blockchain networks

Wouter Joosen & Tom Van Cutsem
DistriNet KU Leuven
November 2023

Learning resources

- Blockchain networks are not covered in the CDK5 handbook
- Instead, recommended background reading:
  - Narayanan *et al.* “Bitcoin and Cryptocurrency Technologies” Princeton University Press - available for free online at: https://bitcoinbook.cs.princeton.edu/
    - Chapter 1 - intro to cryptography and cryptocurrencies
    - Chapter 2 - how Bitcoin achieves decentralization



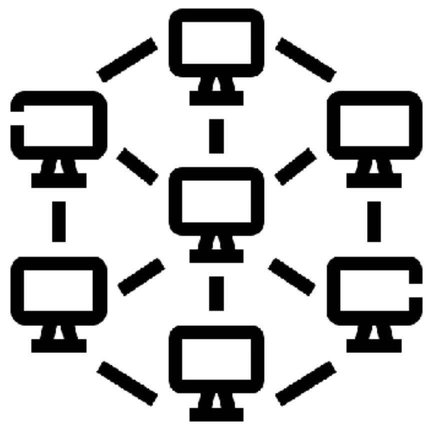
Decentralized systems and blockchain networks

- Centralised vs **decentralized** distributed systems
- Why is a blockchain needed? Example: electronic cash and **the double spending problem**
- How are **transactions** processed in a blockchain network?
- How is **cryptography** used to securely record transactions on a blockchain?
- How is **consensus** achieved in a blockchain network in the face of **sybil attacks**?
- “**Permissioned**” versus “**permissionless**” blockchains

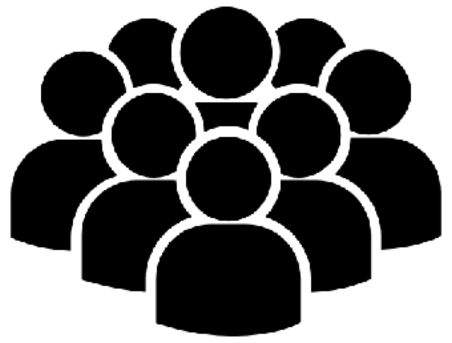
Decentralized Systems: introduction

What does decentralisation mean?

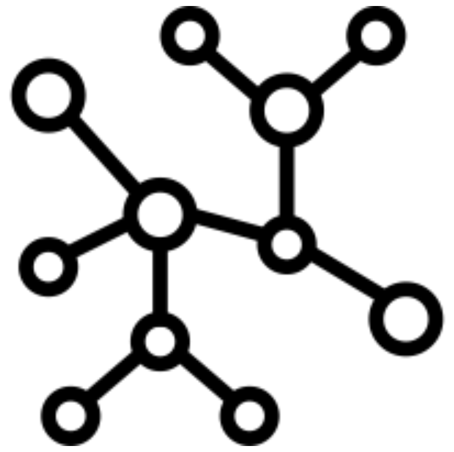
- What do we mean when we say a system is “decentralised”?



- **Architectural (de)centralisation** — how many physical computers is a system made up of? How many of those computers can it tolerate breaking down at any single time?



- **Political (de)centralisation** — how many individuals or organizations ultimately control the computers that the system is made up of? Who sets the rules?



- **Logical (de)centralisation**— does the interface and data structures that the system presents and maintains look more like a single monolithic object, or an amorphous swarm? One simple heuristic is: if you cut the system in half, including both providers and users, will both halves continue to fully operate as independent units?

(Based on: Vitalik Buterin, “The meaning of Decentralisation”, blog post on medium.com, 2017)

What does decentralisation mean?

Logically centralised

	Politically centralised	Politically decentralised
Architecturally centralised	A traditional database or web server	
Architecturally decentralised	Distributed databases *Permissioned* blockchain networks	*Permissionless* blockchain networks

Logically decentralised

	Politically centralised	Politically decentralised
Architecturally centralised		
Architecturally decentralised	Content Delivery Networks (CDNs)	BitTorrent

(Based on: Vitalik Buterin, “The meaning of Decentralisation”, blog post on medium.com, 2017)

Blockchain networks: examples

- Examples of **permissionless** blockchain networks:
  - Bitcoin (decentralized payments)
  - Ethereum (decentralized computation)
  - Filecoin (decentralized storage)
  - Helium (decentralized wireless networks)
- Examples of **permissioned** blockchain networks:
  - Hyperledger Fabric
  - Corda
  - Private Ethereum networks ("Enterprise Ethereum")
  - Hyperledger Sawtooth

Bitcoin Ethereum

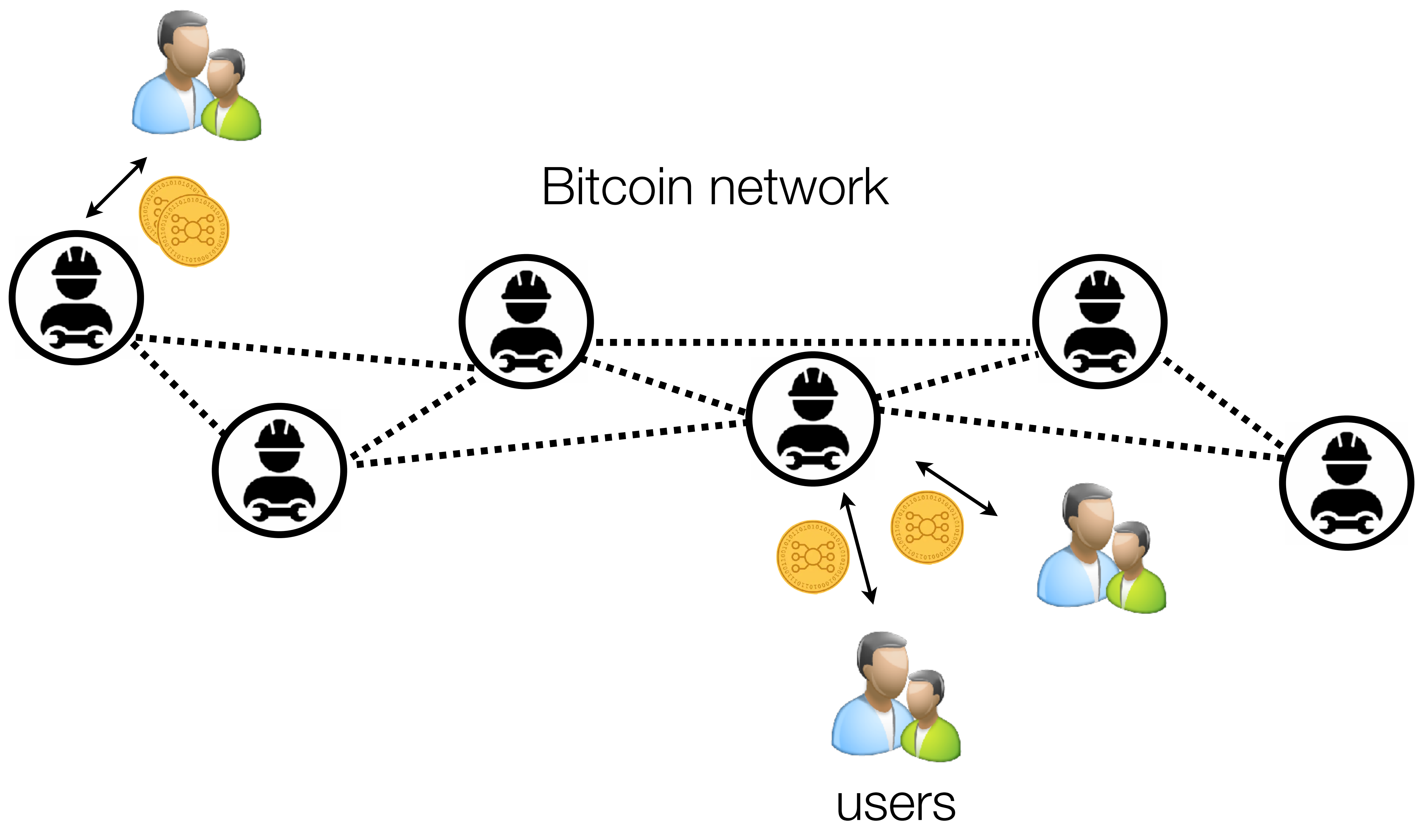
Filecoin Helium

What problem does a blockchain solve?

Example: decentralized payments using Bitcoin

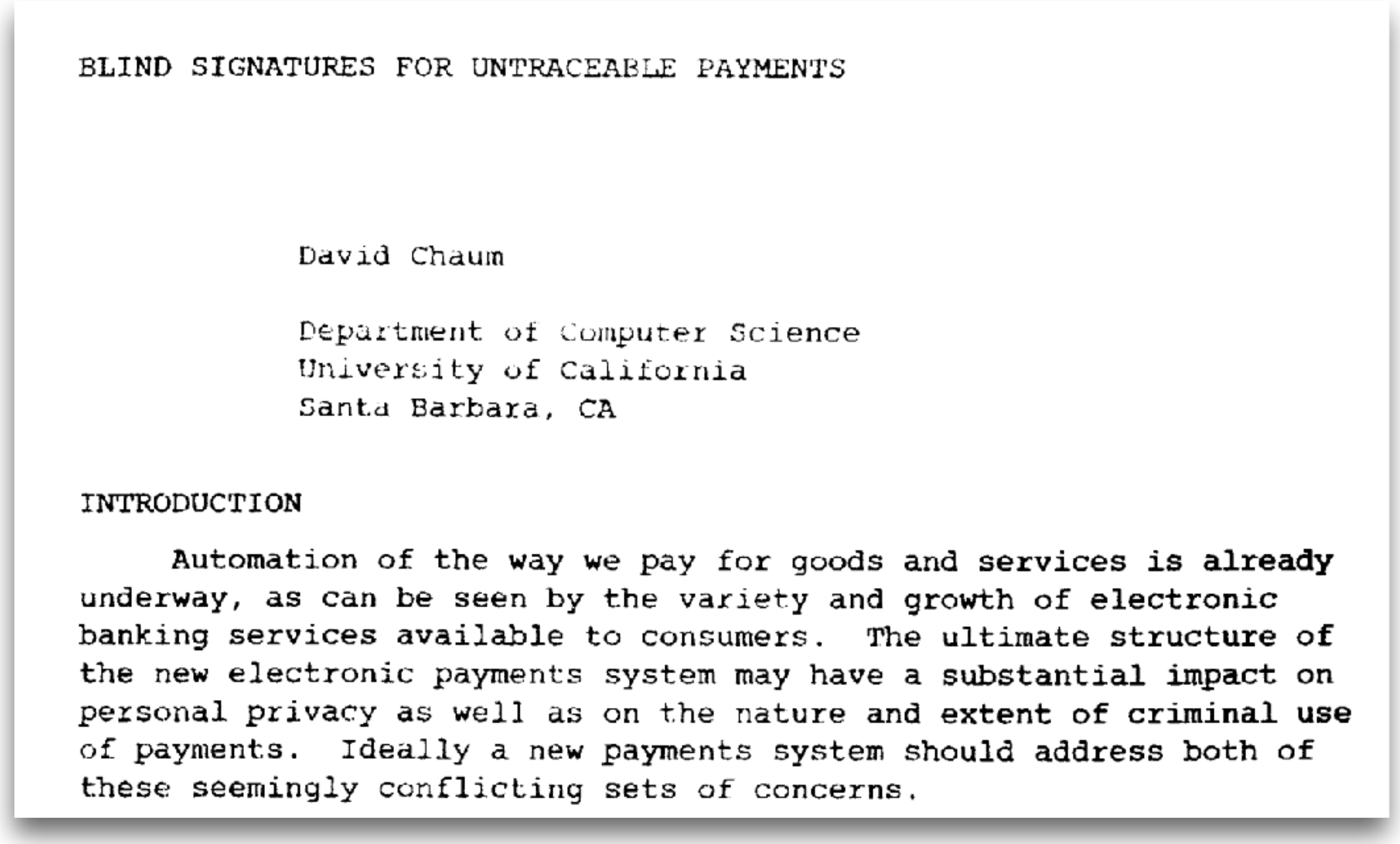
Bitcoin is a decentralised payment network

- Not controlled by any single company or institution
- Introduces its own digital currency unit known as a bitcoin (Bitcoin = the network/protocol, bitcoin = the currency)
- Payment transactions are communicated over a **peer-to-peer** network
- Each **node** in the network **verifies** the validity of each transaction
- Valid transactions become part of a global, replicated, **public ledger**
- The network creates its own **money supply** according to a fixed algorithm
- Users are **pseudonymous**



Electronic **cash**

- Since the dawn of the Internet, cryptographers have tried to create digital currencies that are similar to physical cash or coins, supporting direct person-to-person *anonymous* and *untraceable* payments.
- Money as a "bearer instrument" token: whoever holds the token can spend it
- Early example: e-cash (Digicash)


BLIND SIGNATURES FOR UNTRACEABLE PAYMENTS

David Chaum

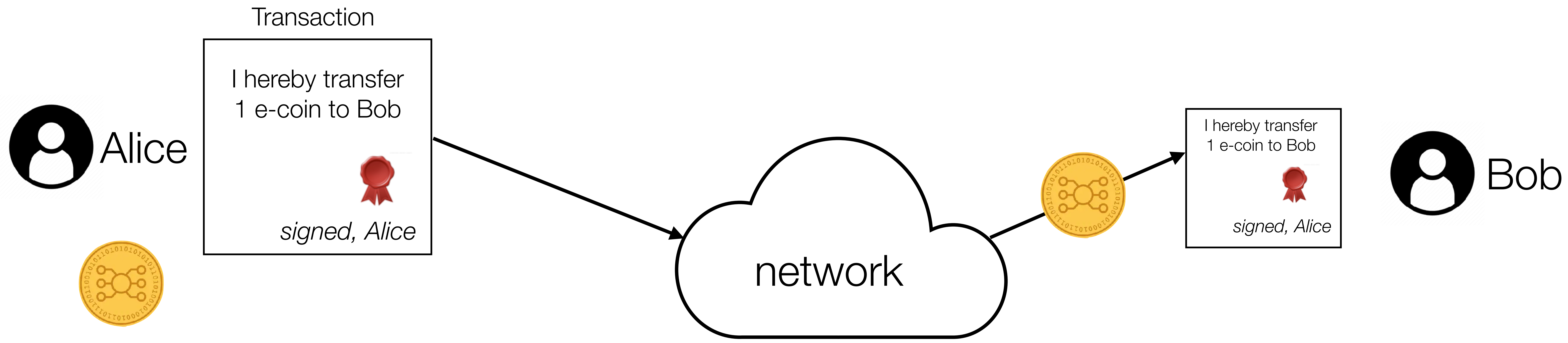
Department of Computer Science
University of California
Santa Barbara, CA

INTRODUCTION

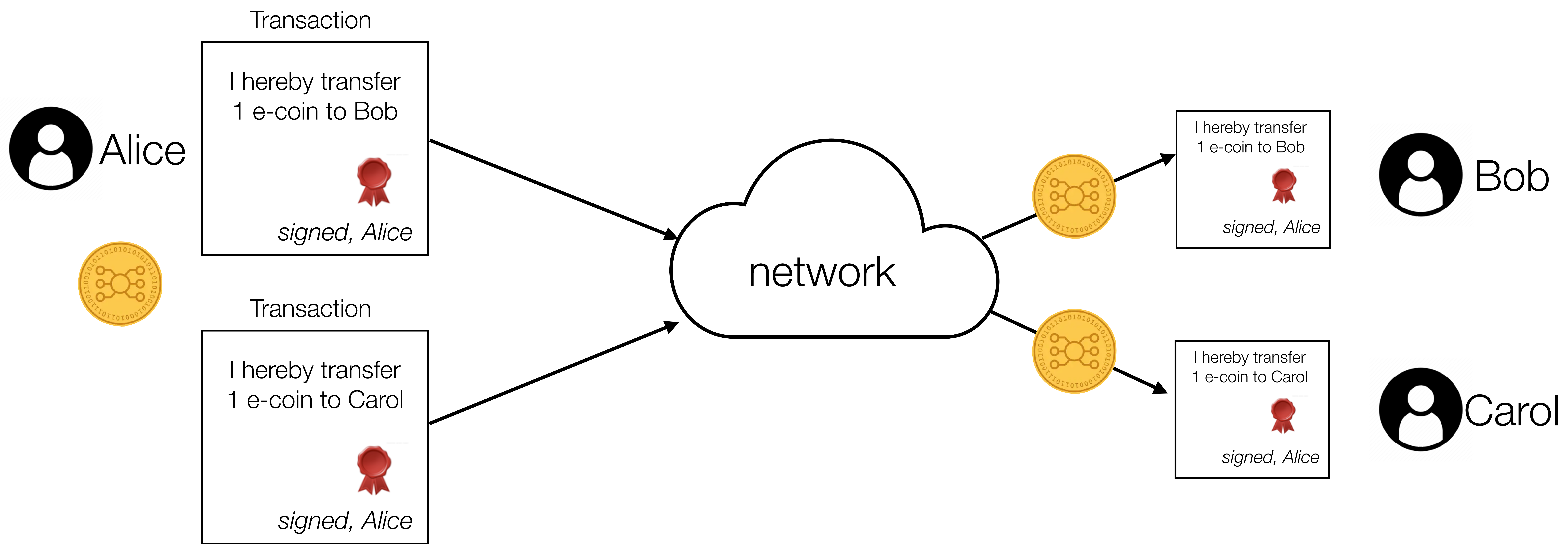
Automation of the way we pay for goods and services is already underway, as can be seen by the variety and growth of electronic banking services available to consumers. The ultimate structure of the new electronic payments system may have a substantial impact on personal privacy as well as on the nature and extent of criminal use of payments. Ideally a new payments system should address both of these seemingly conflicting sets of concerns.

David Chaum
Electronic cash (1982)

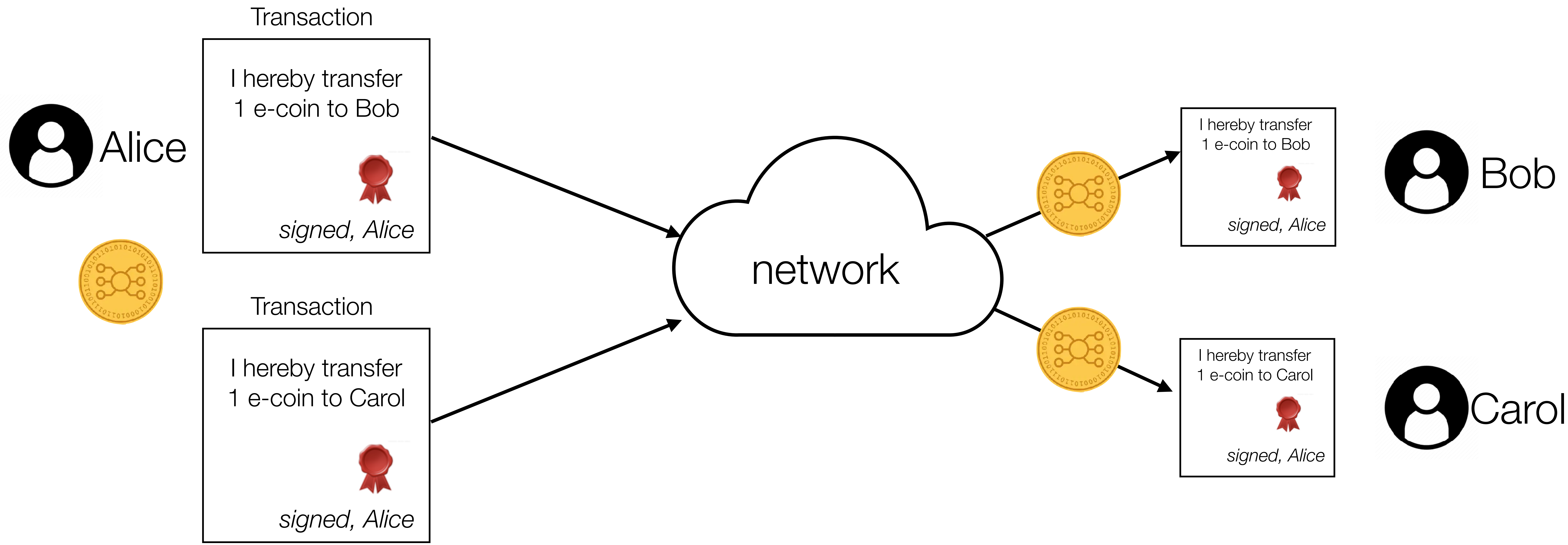
The problem with electronic cash: the Double Spending Problem



The problem with electronic cash: the Double Spending Problem



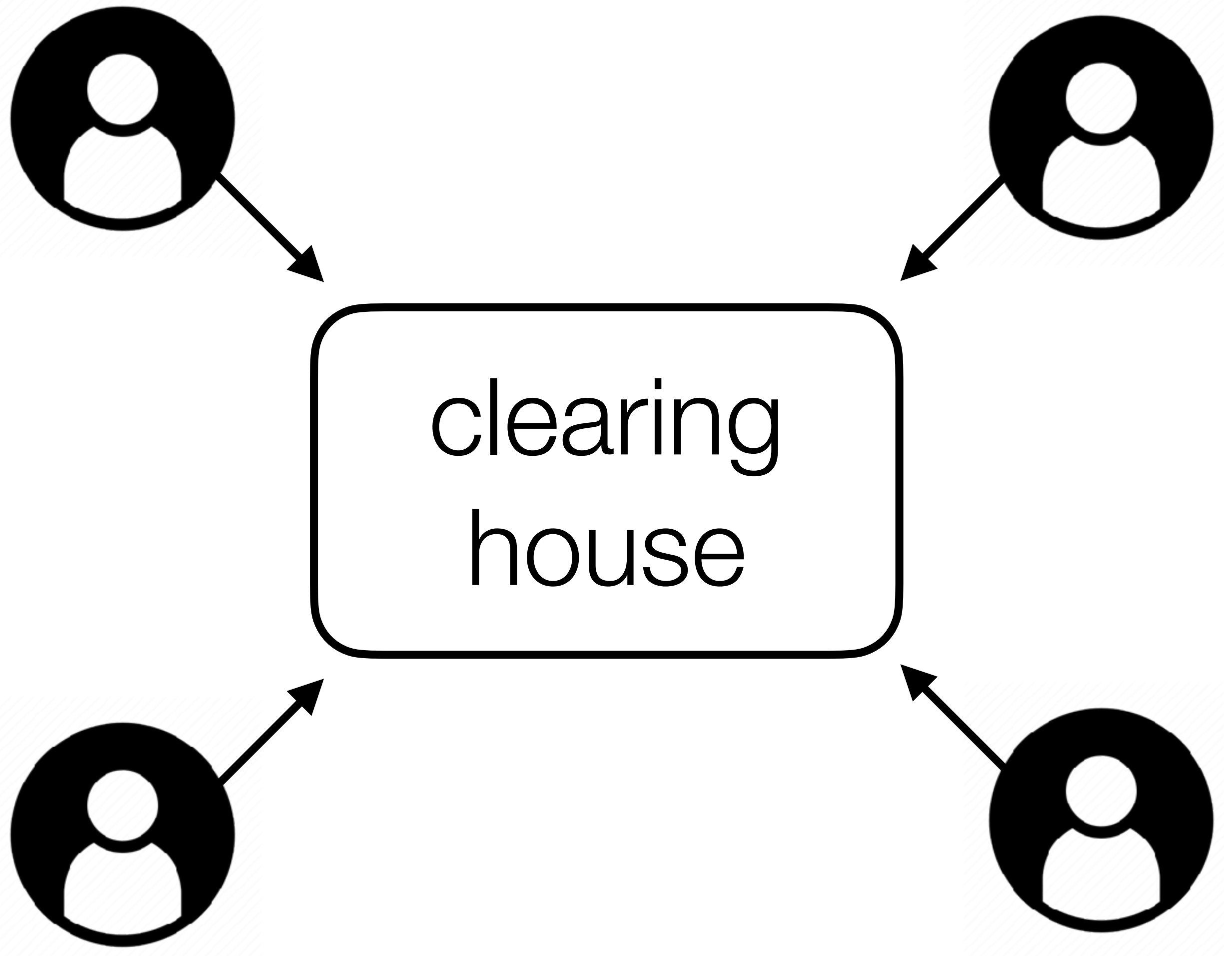
The problem with electronic cash: the Double Spending Problem



How can Bob and Carol be sure they are now **the sole owner** of Alice's coin?

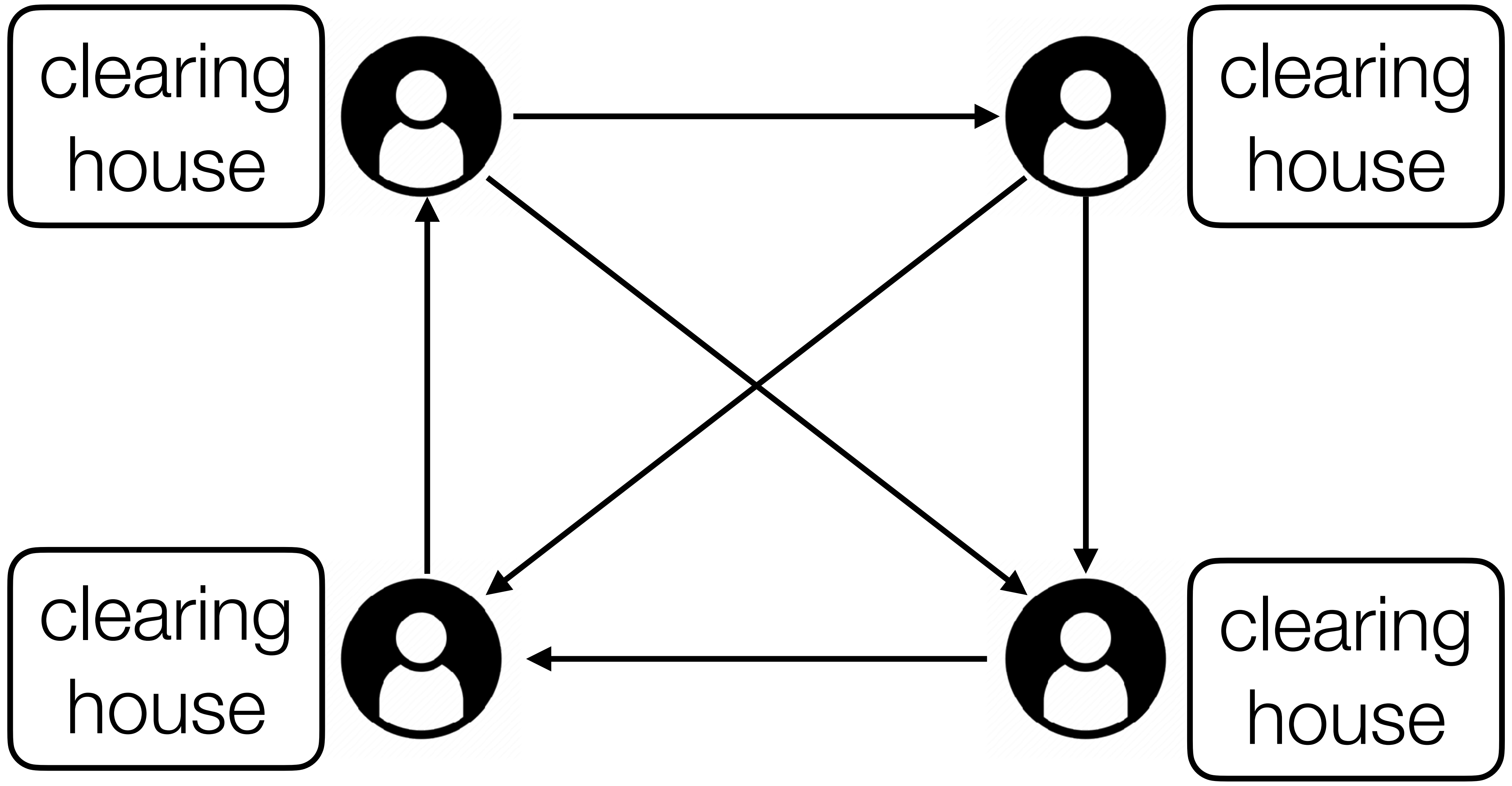
Straightforward solution: use a central clearing house

- The clearing house does the accounting of what tokens have already been spent. This **avoids "double spending"** the same token.
- The payments themselves can still be **anonymous**! We just need to record spent tokens. Privacy risks can be partially mitigated using "blind signatures".
- Problem: everyone **depends** on the clearing house. **Risks**:
  - **Technical** risks: availability (what if the clearing house is unavailable?) and security (what if the clearing house gets attacked? This may include insider threats!)
  - **Economic** and **political** risks: what if the company running the clearing house goes bankrupt or is threatened in court? (E.g. Digicash actually went bankrupt in 1998)

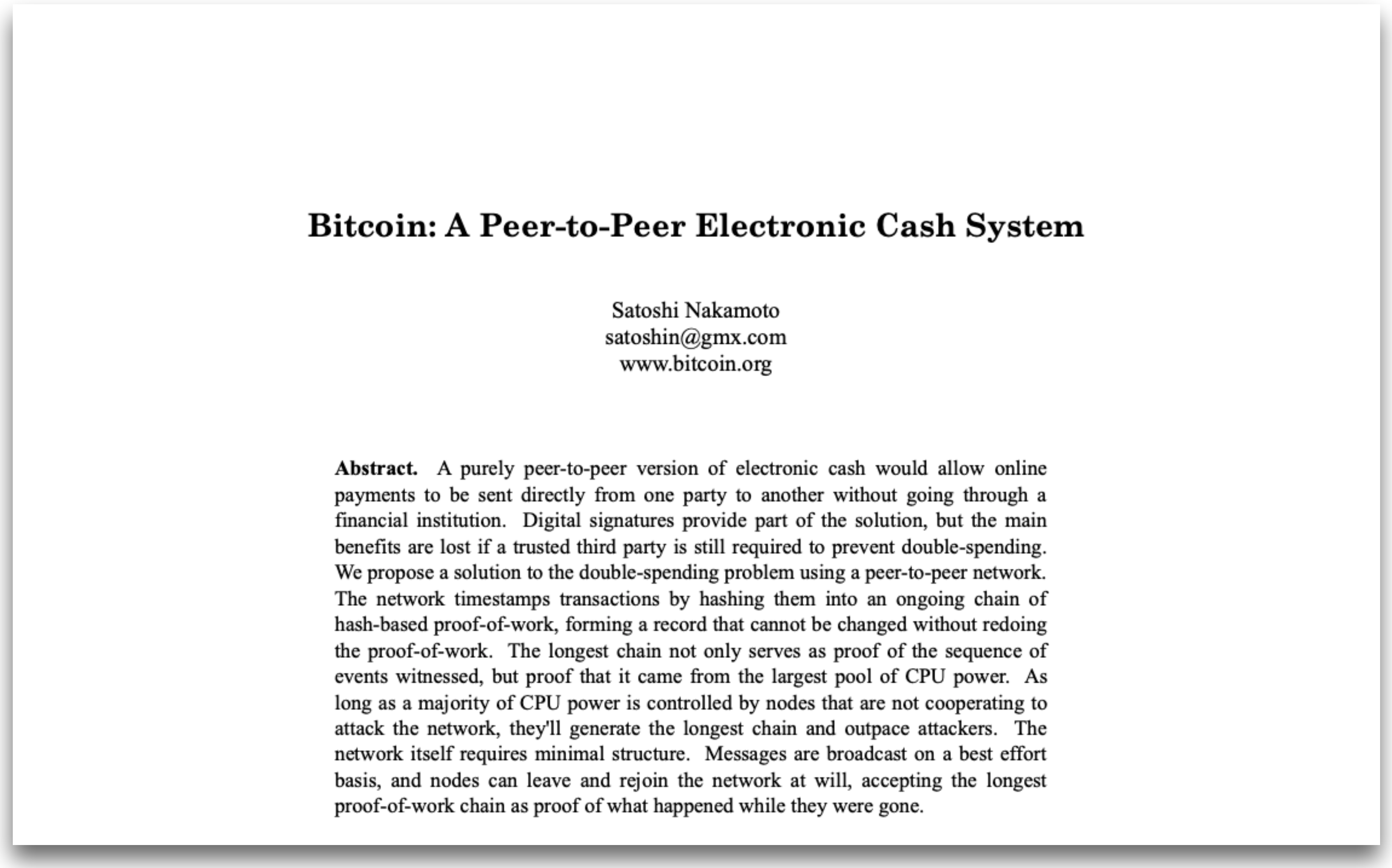


Blockchain networks

- Bitcoin’s breakthrough idea: rather than having a single party record who owns what, let ***everyone and anyone*** collectively do the accounting of who owns what
- Store payment transactions in an append-only **replicated database** called a **blockchain**



Fully **decentralised** payment network



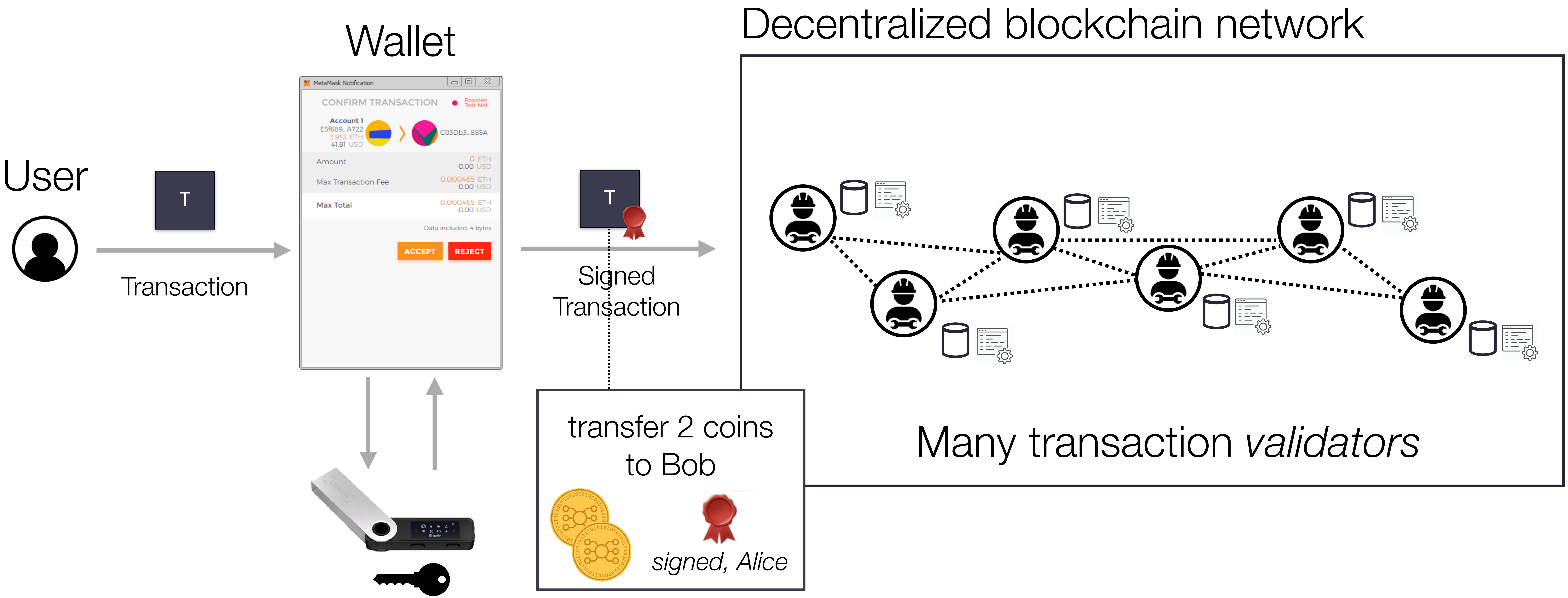
Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

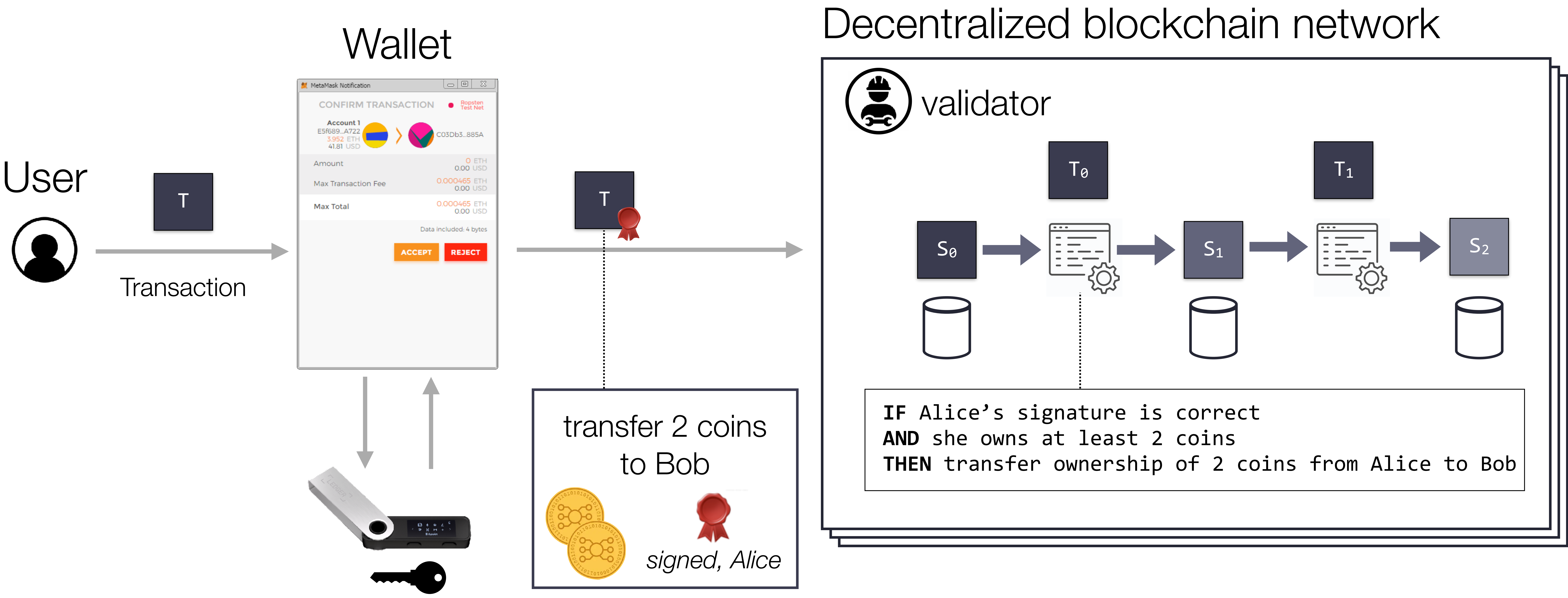
Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

The “Bitcoin whitepaper”
by “Satoshi Nakamoto”, 2008
https://bitcoin.org/bitcoin.pdf

Payment transactions in a blockchain network



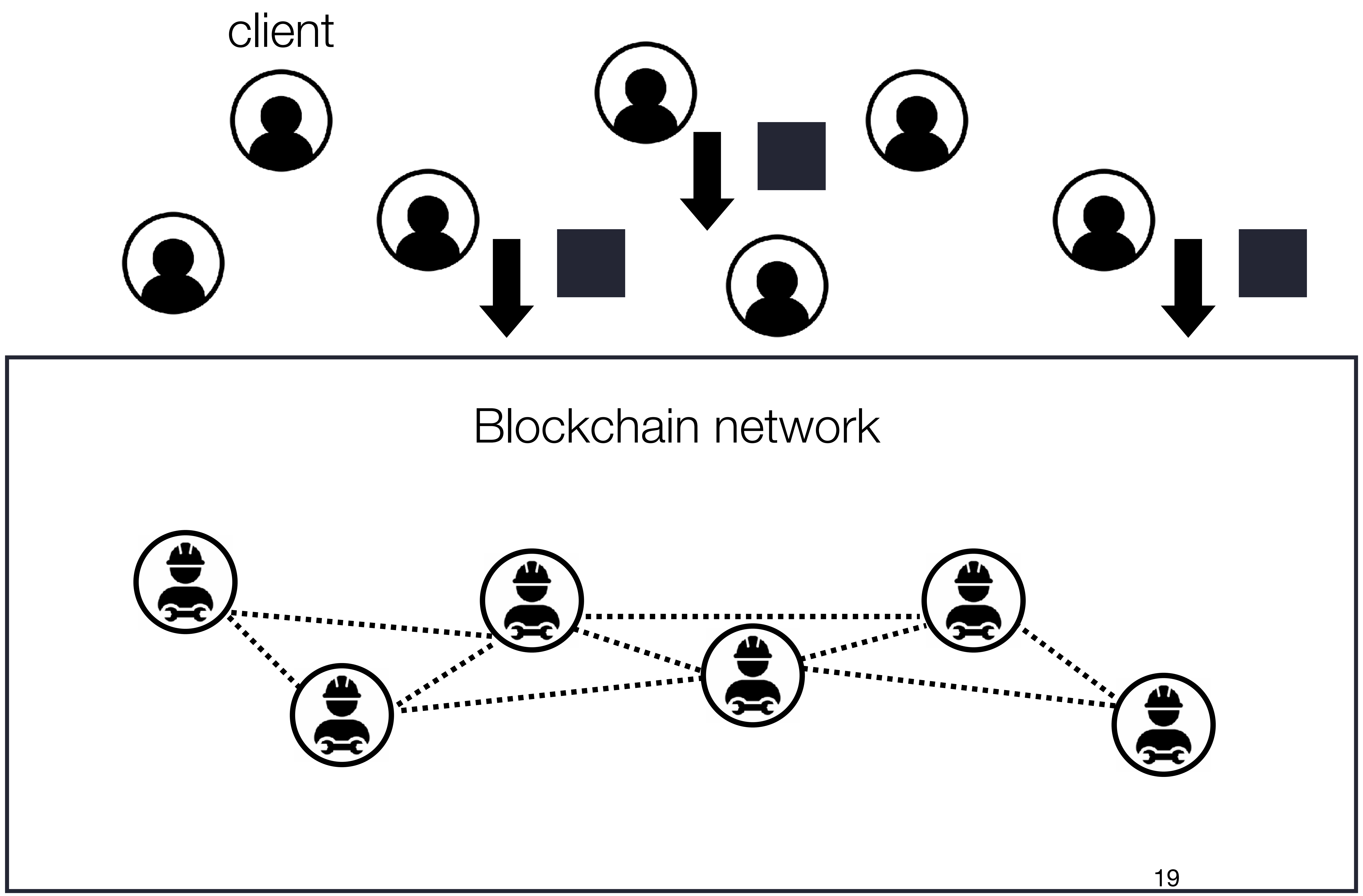
Blockchain networks are replicated state machines!



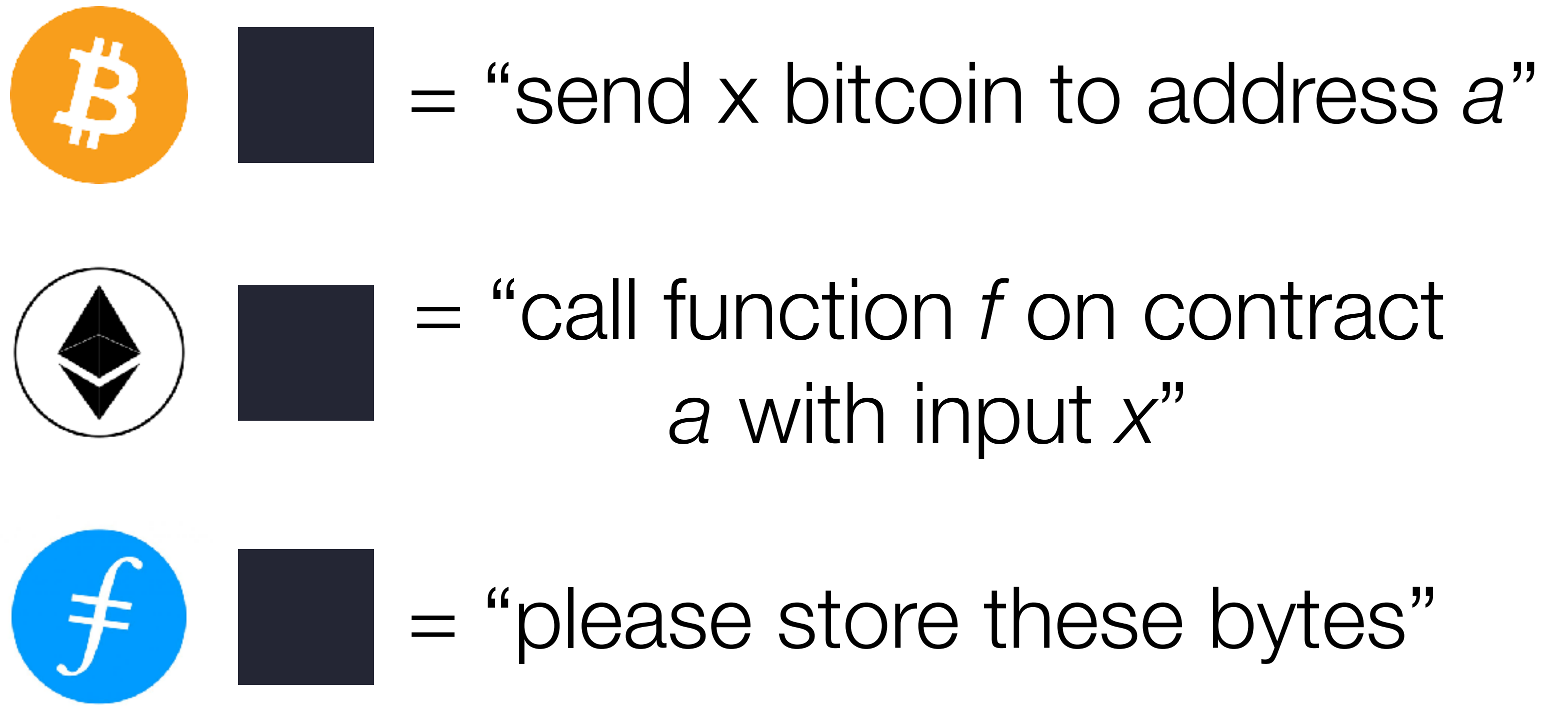
How does a Blockchain network process transactions?

Step 1: clients submit signed transactions

- Clients **concurrently** submit signed transactions to one or more validators.

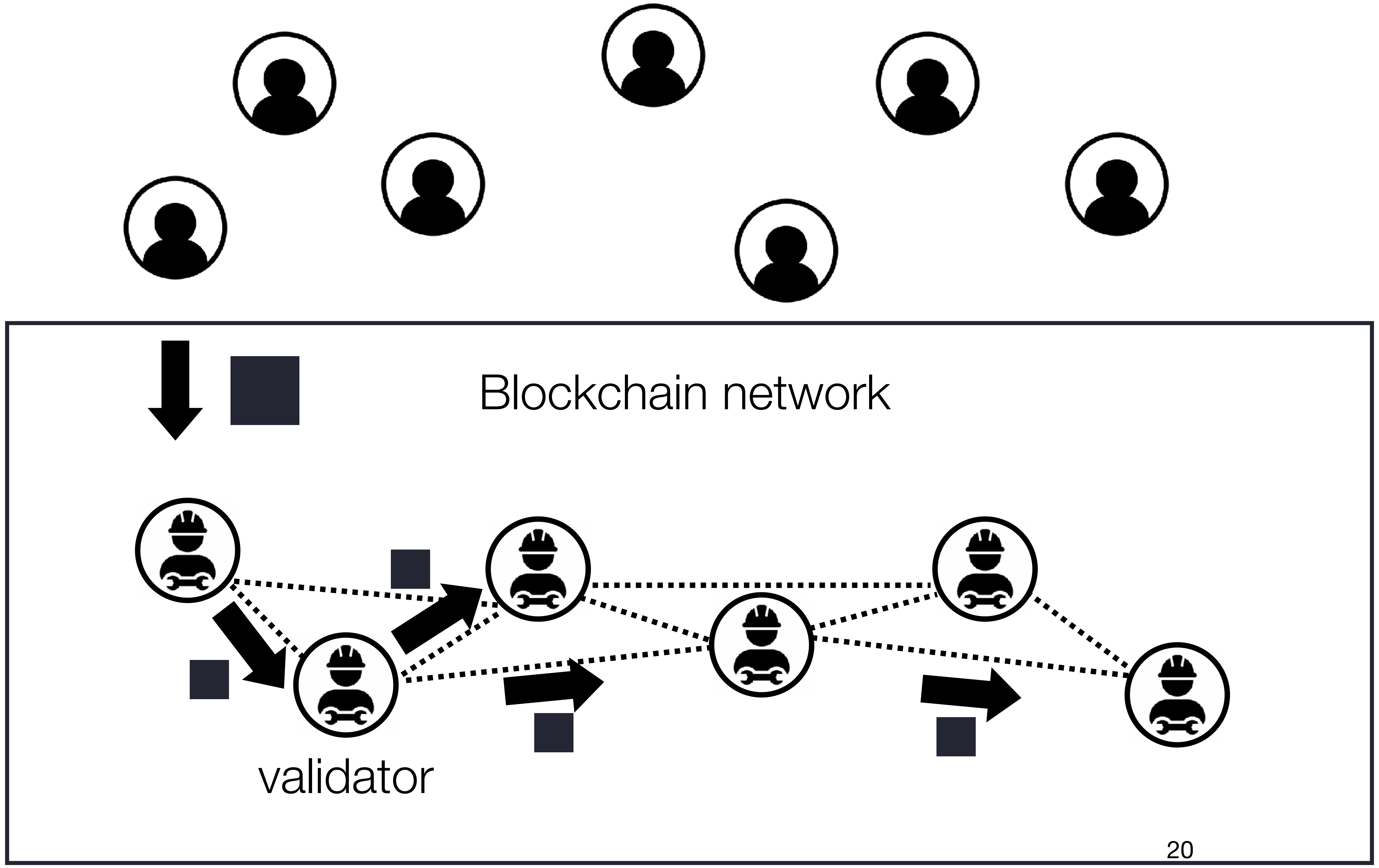


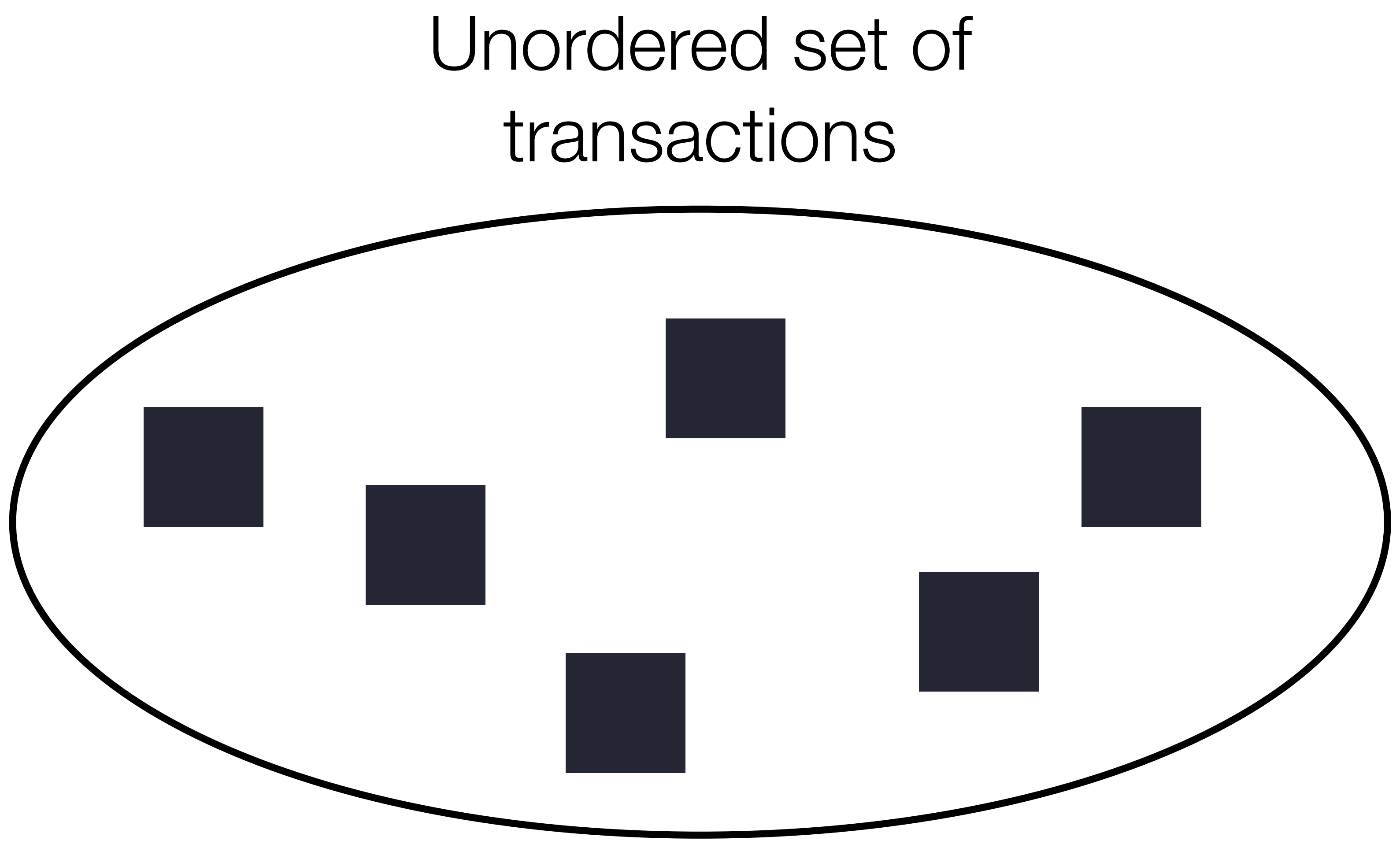
Example transactions…



Step 2: validators validate and gossip transactions

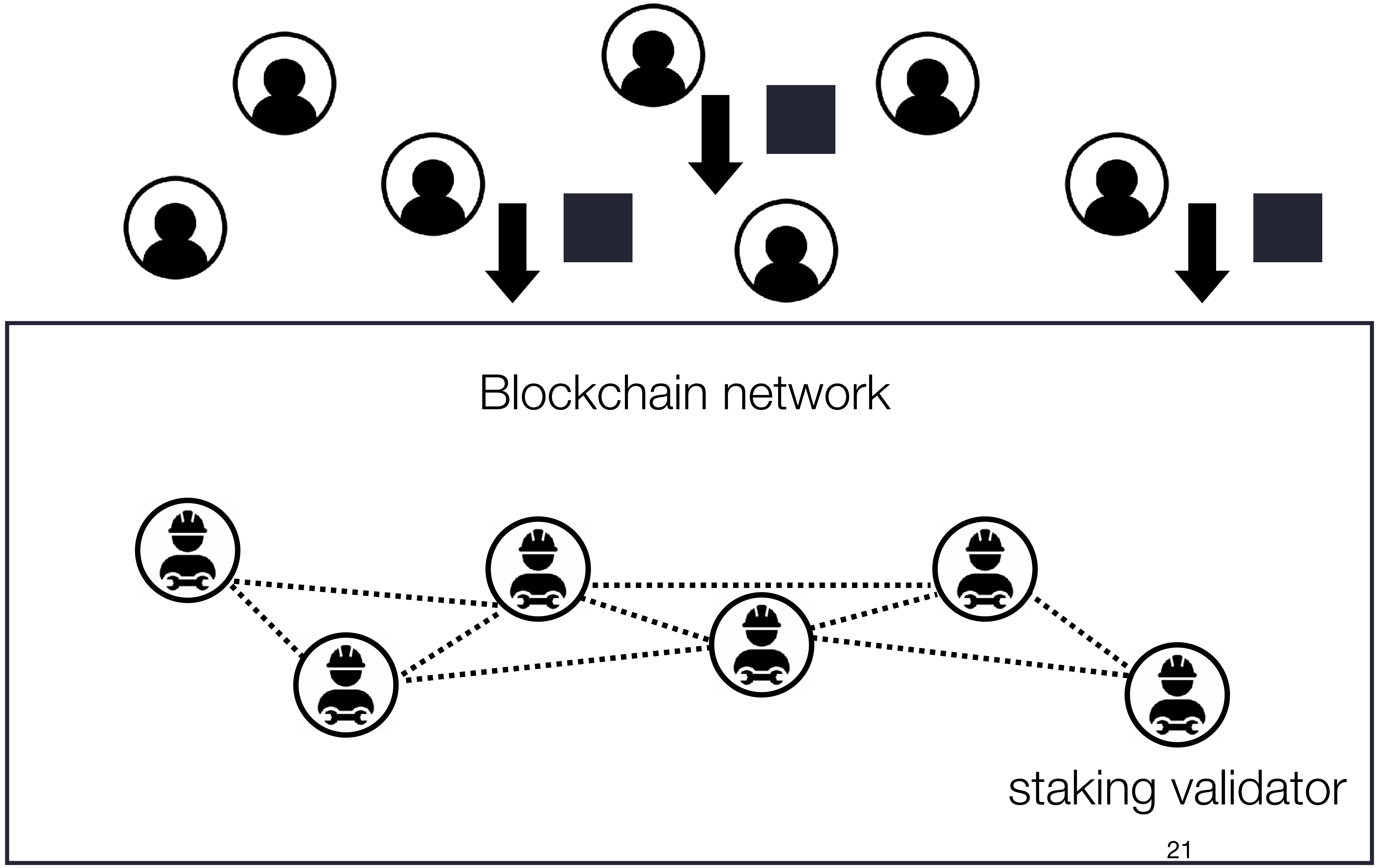
- A **validator** is a network node that maintains an **unordered set** ("mempool") of incoming transactions. It collects, validates and broadcasts transactions to other peers (using a **gossip** broadcast protocol)

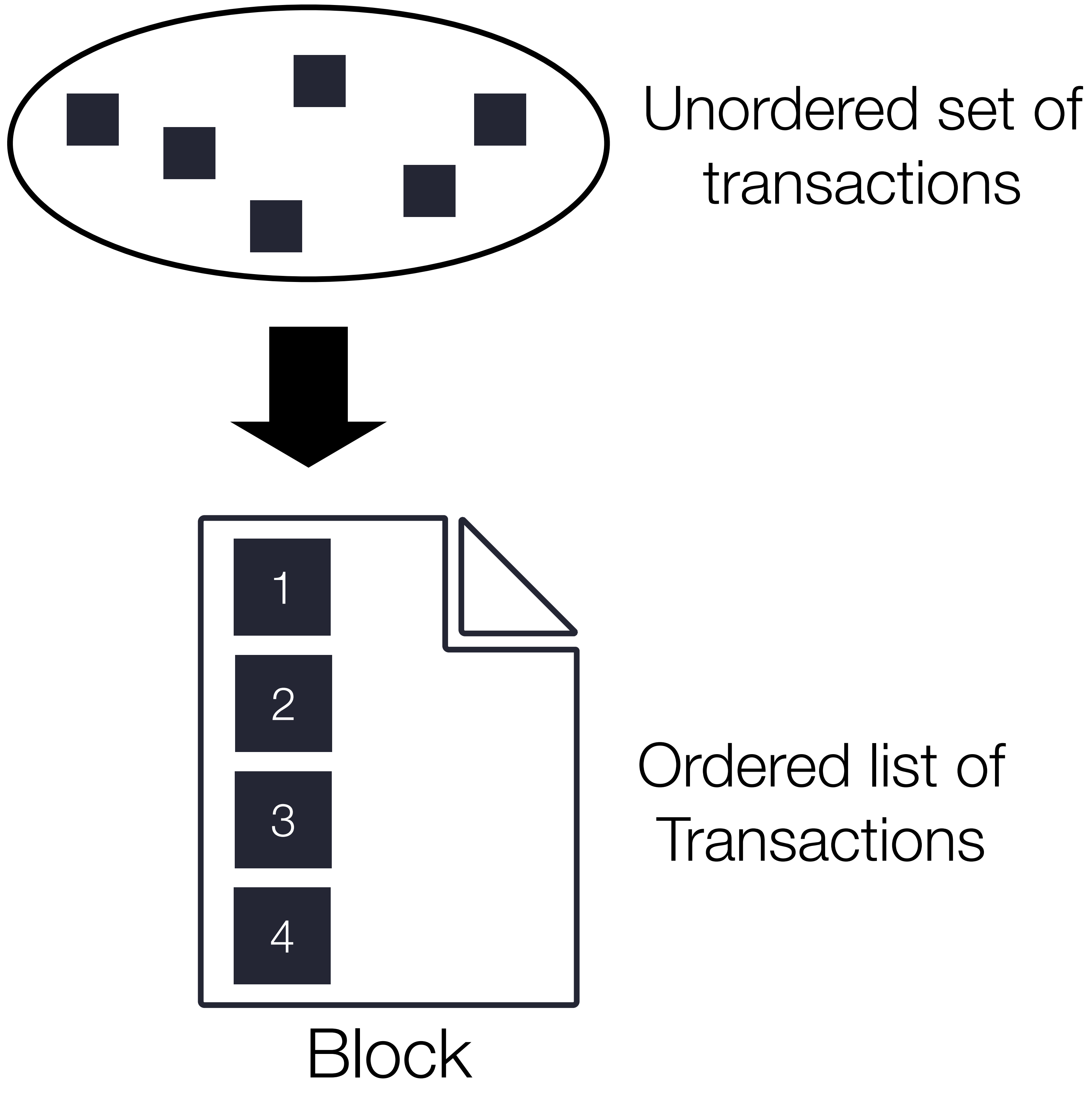




Step 3: a validator produces a block of transactions

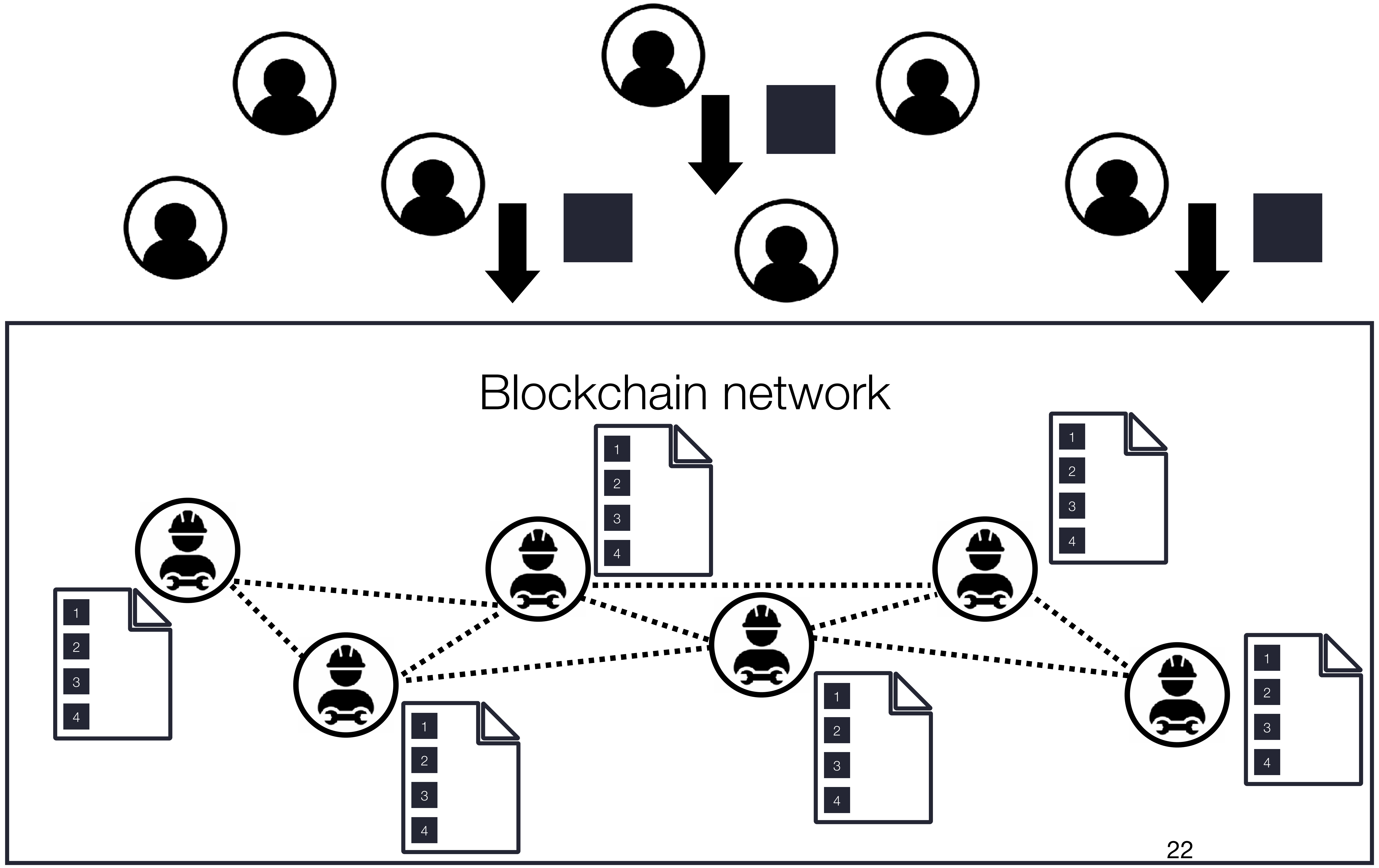
- At regular intervals, a subset of validators pick a subset of transactions from the pool and *sequence* them, thus producing an *ordered* list of transactions. These validators are sometimes called “**miners**” or “**staking validators**”. The transaction list is called a “block”

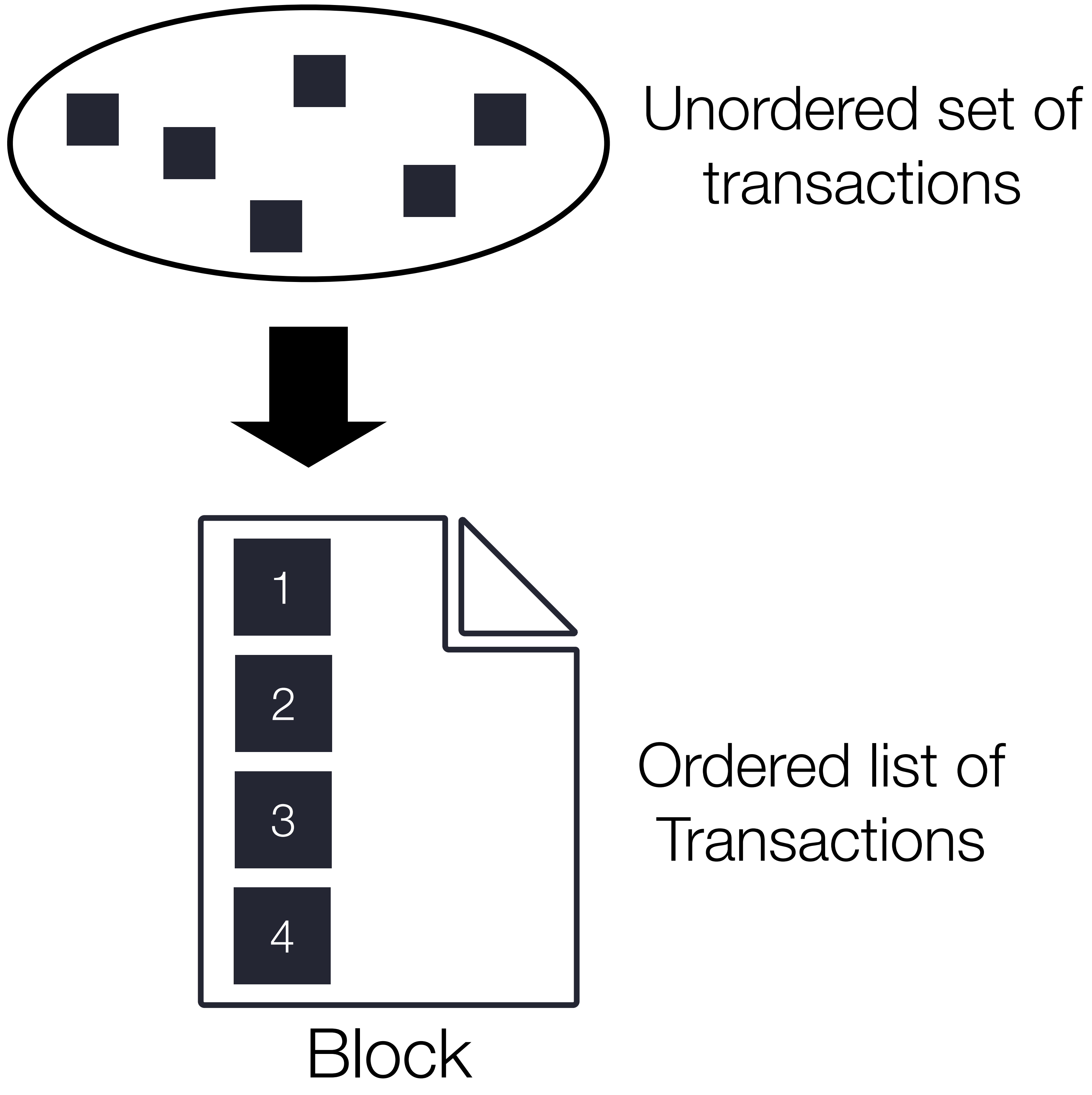




Step 4: validators gossip block and append to the blockchain

- The **block is broadcast** to all validators (again using gossip). Each validator **checks again** if all transactions in the block are valid. If yes, they **append** the block to their local transaction log (aka the blockchain).

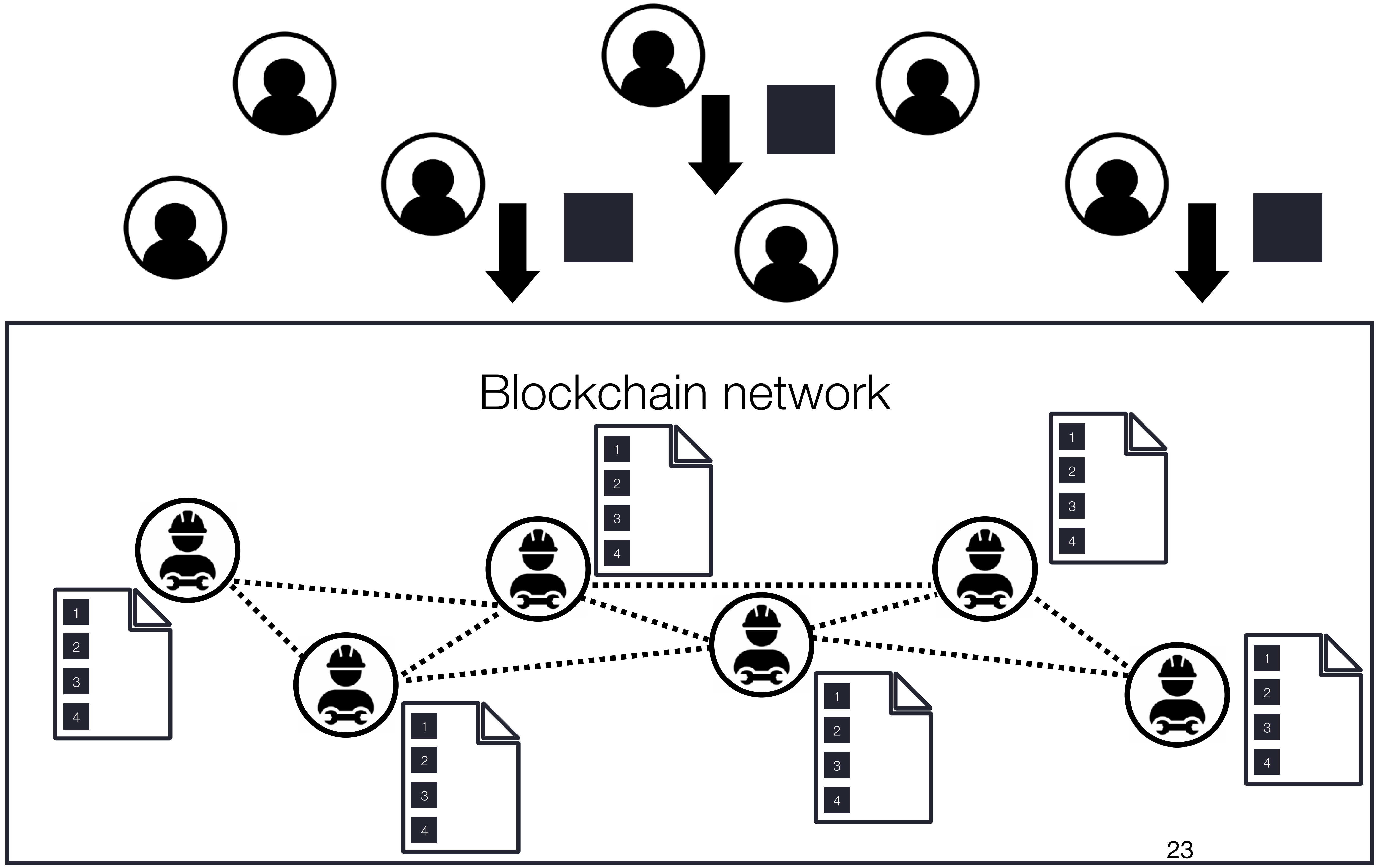


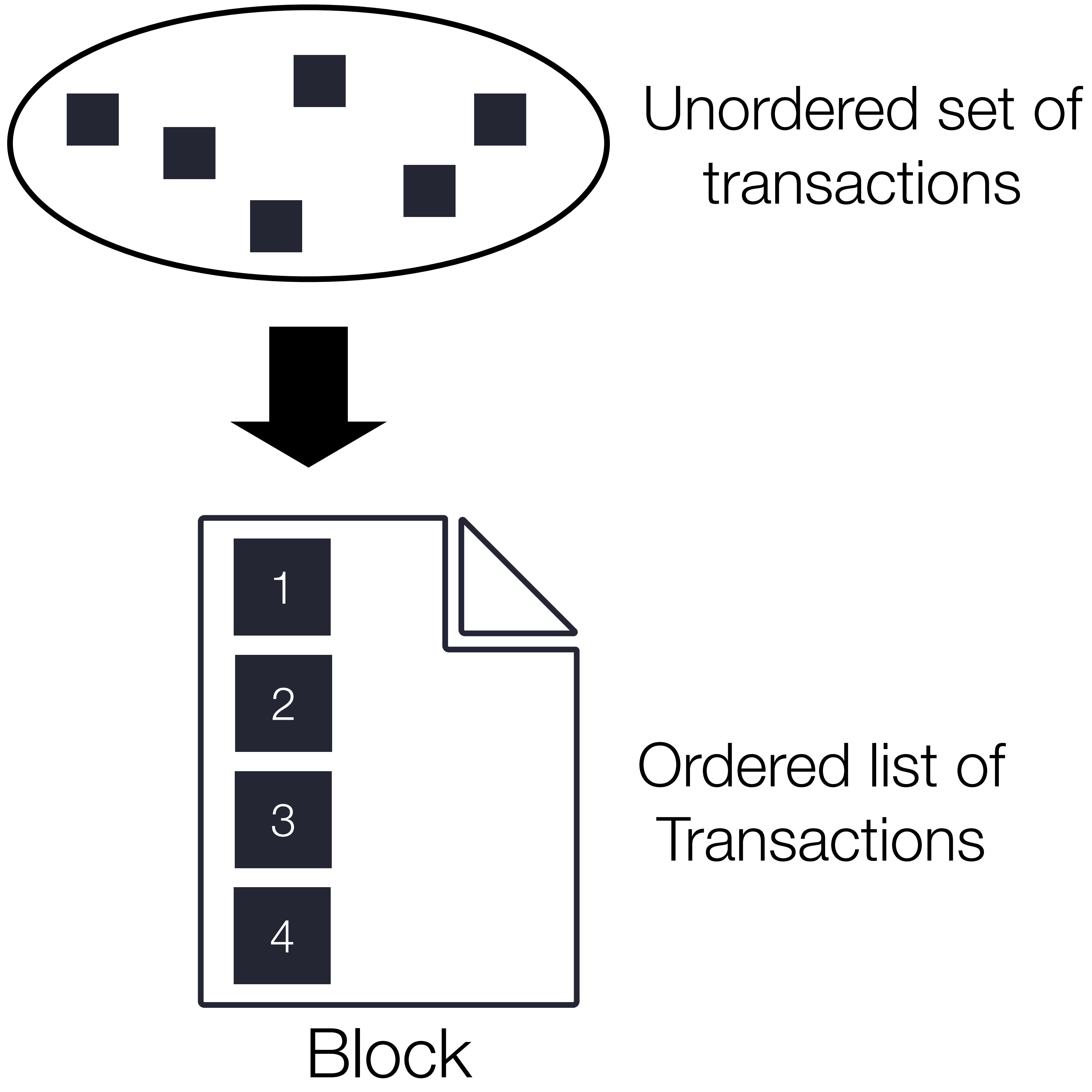


KU LEUVEN DistriNet

Consensus

- All validators must reach **consensus** on the exact same transaction history!
- Need to make sure that blocks get appended everywhere *in the same order*

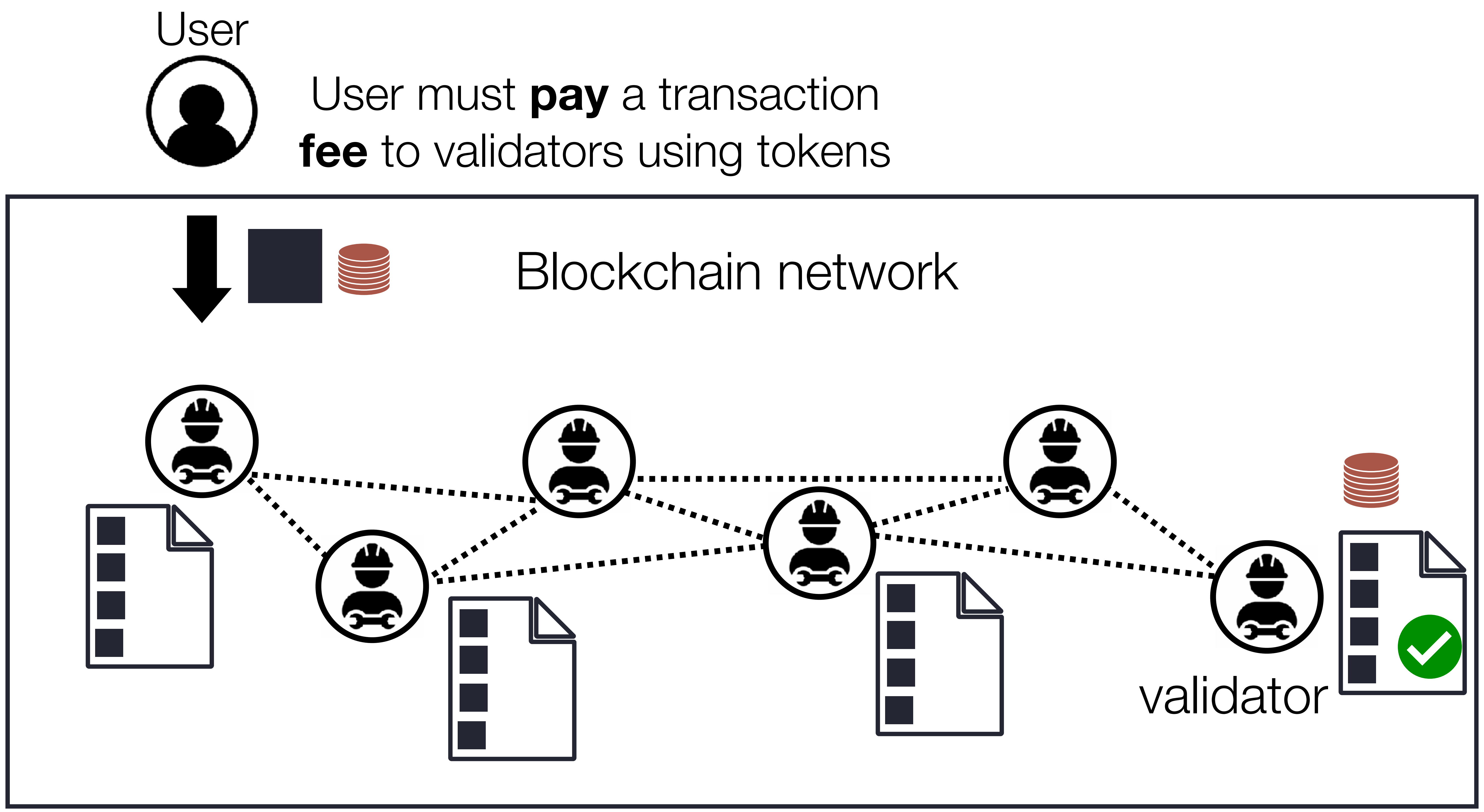




KU LEUVEN DistriNet

Blockchain networks: tokens, transaction fees and mining rewards

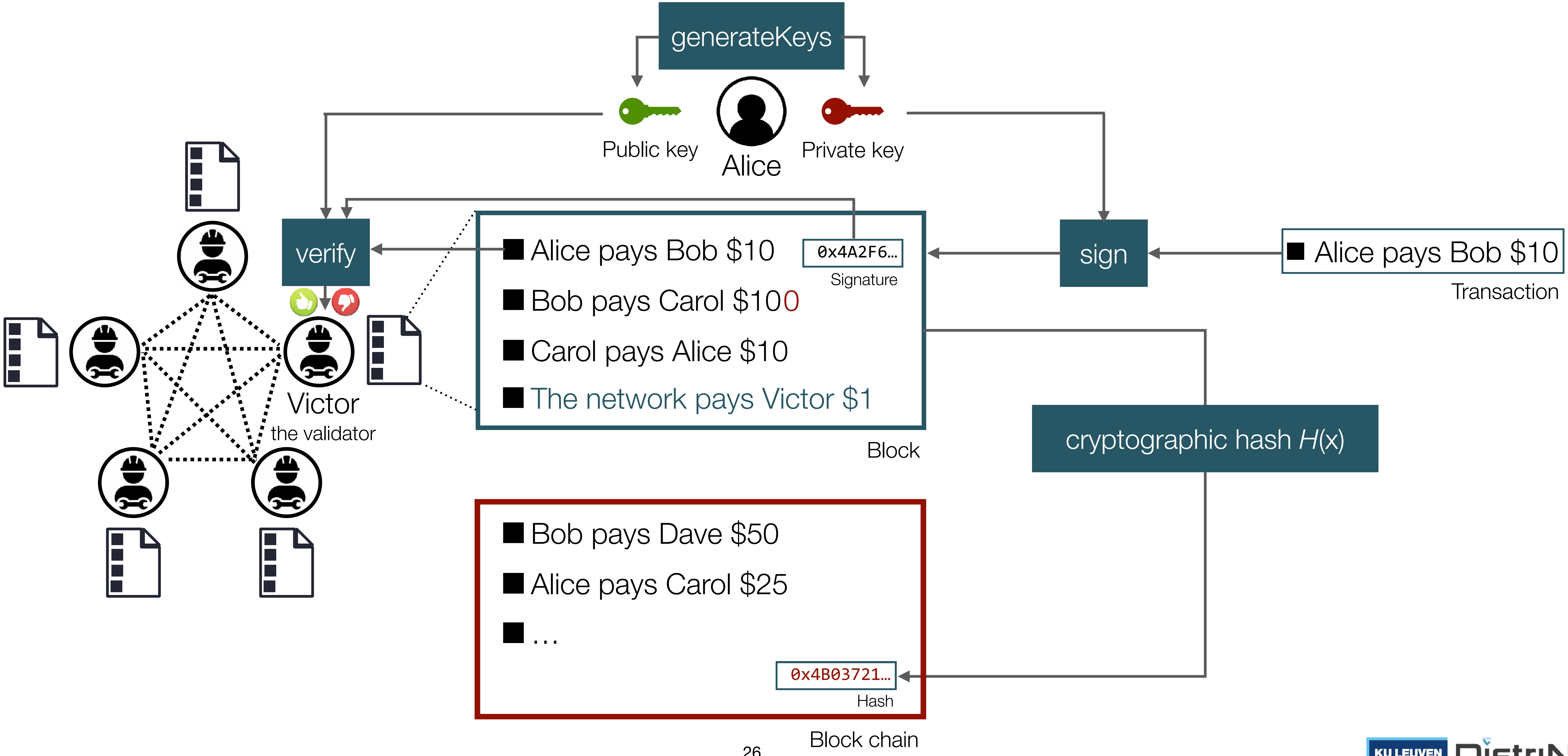
- **Tokens** are used to a) pay for transaction processing (transaction **fee**) and b) to **reward** validators for contributing hardware resources (compute, bandwidth, storage) to validate transactions. They act as an **incentive mechanism** to keep validators honest.



Validators can **earn** additional tokens by producing valid blocks (a process called “mining” or “staking”)

What are the cryptographic building blocks of a blockchain?

How cryptography is used to securely record transactions on a blockchain

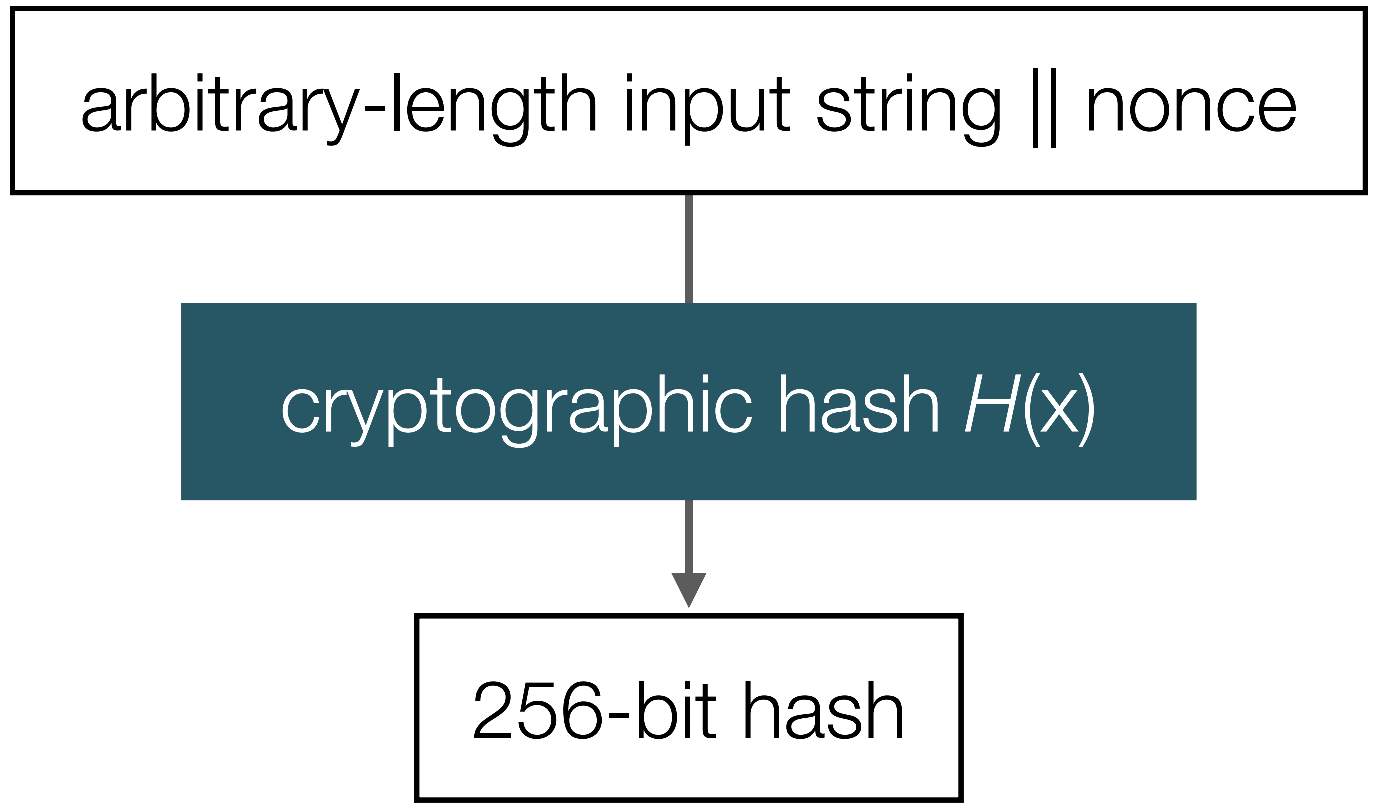


Common cryptographic algorithms used in blockchain systems

Cryptographic hashes

SHA-256

Secure hash algorithm



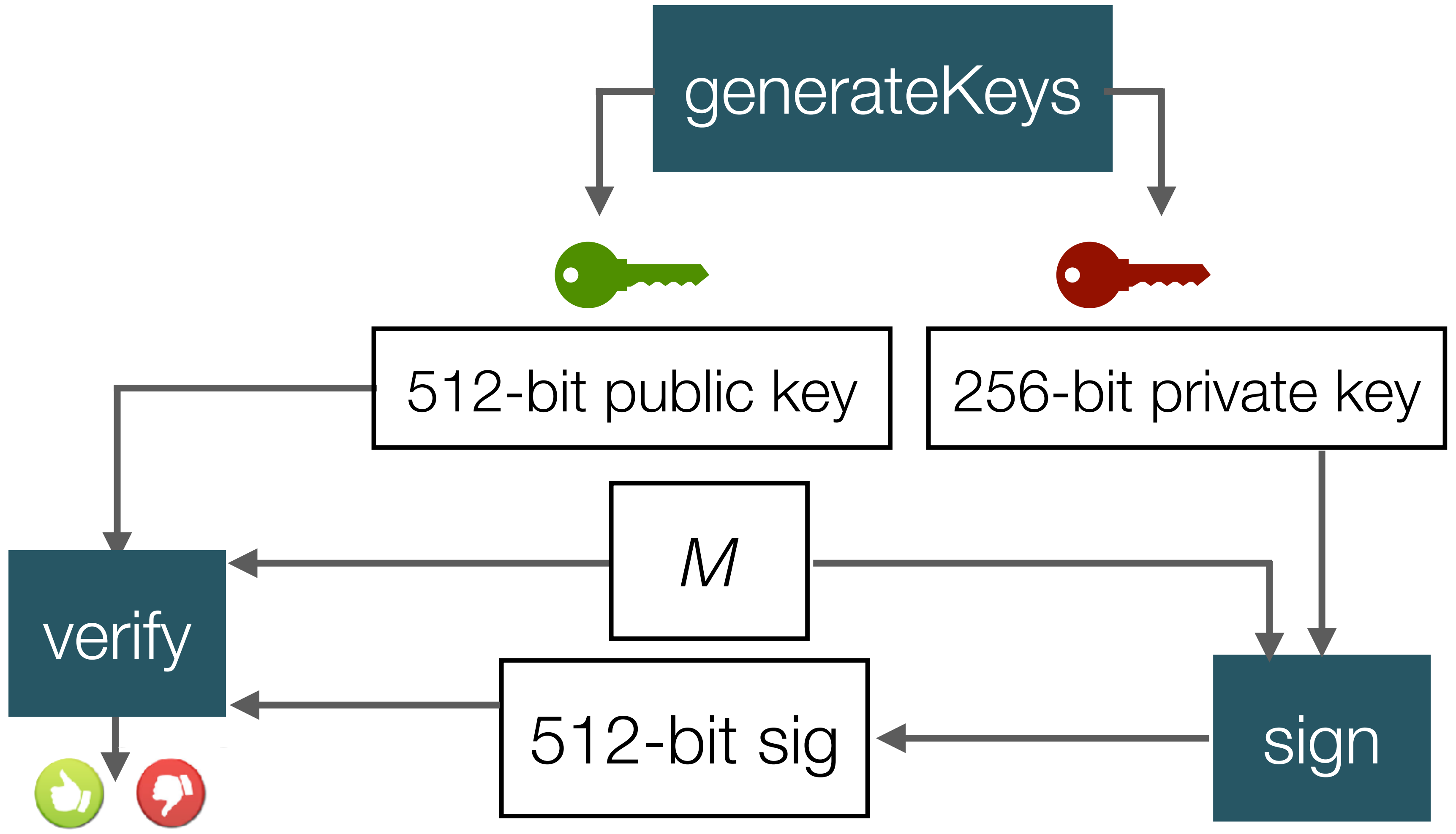
Desirable properties:

- *H* is collision-resistant
- *H* hides its input x
- *H* is “puzzle-friendly”

Digital signatures

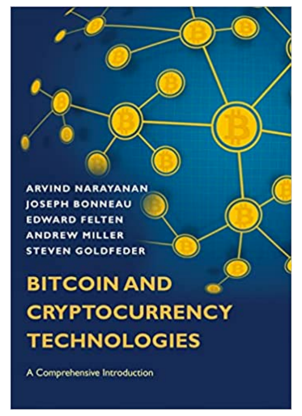
ECDSA

Elliptic curve digital signature algorithm



Desirable properties:

- Valid signatures must verify
- Signatures are unforgeable
- Signature is unique to *M*

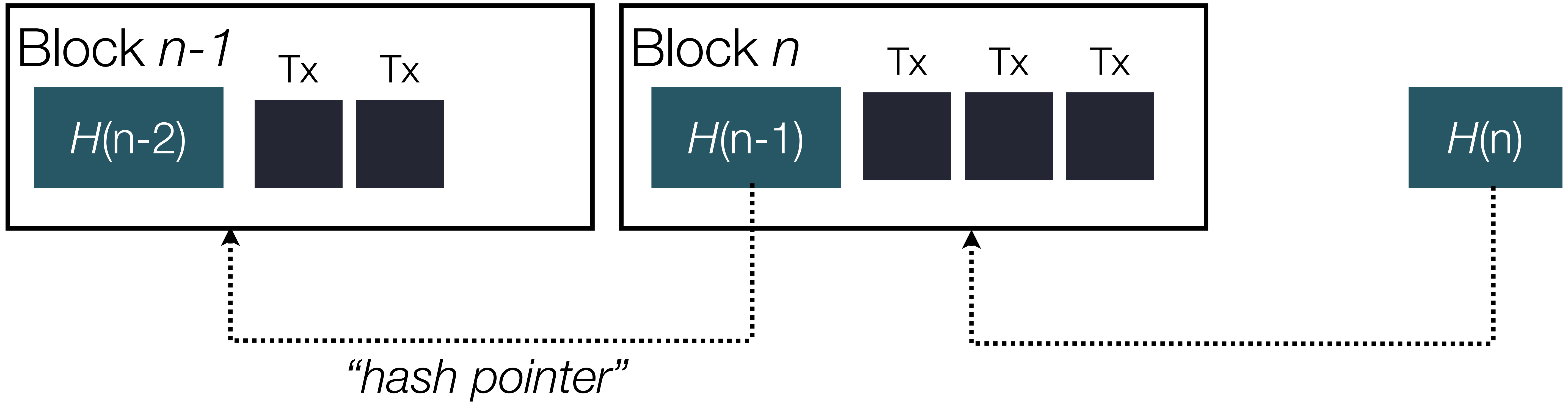


See book chapter 1 for details

KU LEUVEN DistriNet

Common cryptographic algorithms used in blockchain systems

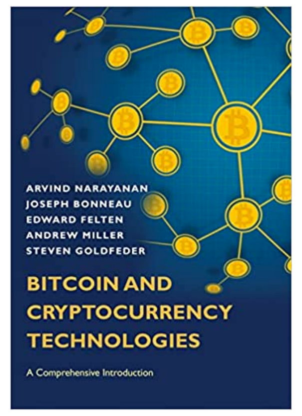
Hash pointers



The hash is used both as:

- a unique identifier (to identify and lookup the data)
- a digest (to verify that the data has not been tampered with)

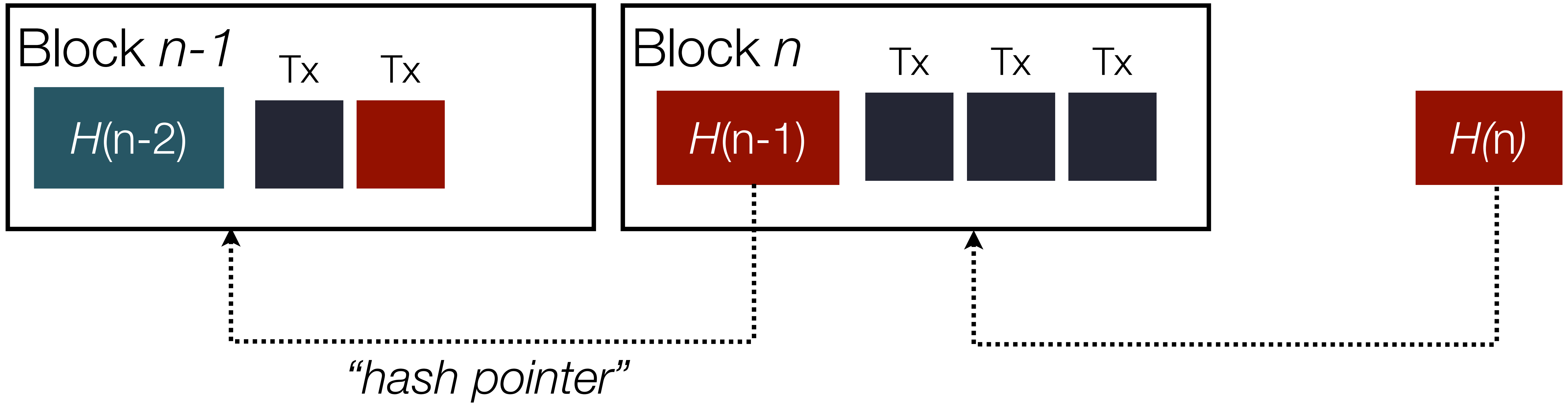
Any non-cyclical data structure can be built from hash pointers



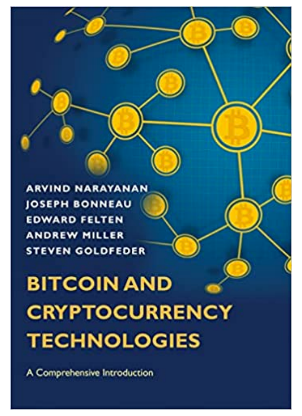
See book chapter 1 for details

Common cryptographic algorithms used in blockchain systems

Why use hash pointers?



- We want the transaction log history to be immutable (i.e. only ***append*** new transactions, not ***edit*** past transactions).
- By using hash pointers, we ensure that modifying *any* data in *any* past block would **invalidate** the hash pointers of *all* the following blocks.
- This makes it immediately clear to anyone with a historical copy of the blockchain that data has been tampered with.
- This makes the transaction log "**tamper-evident**".



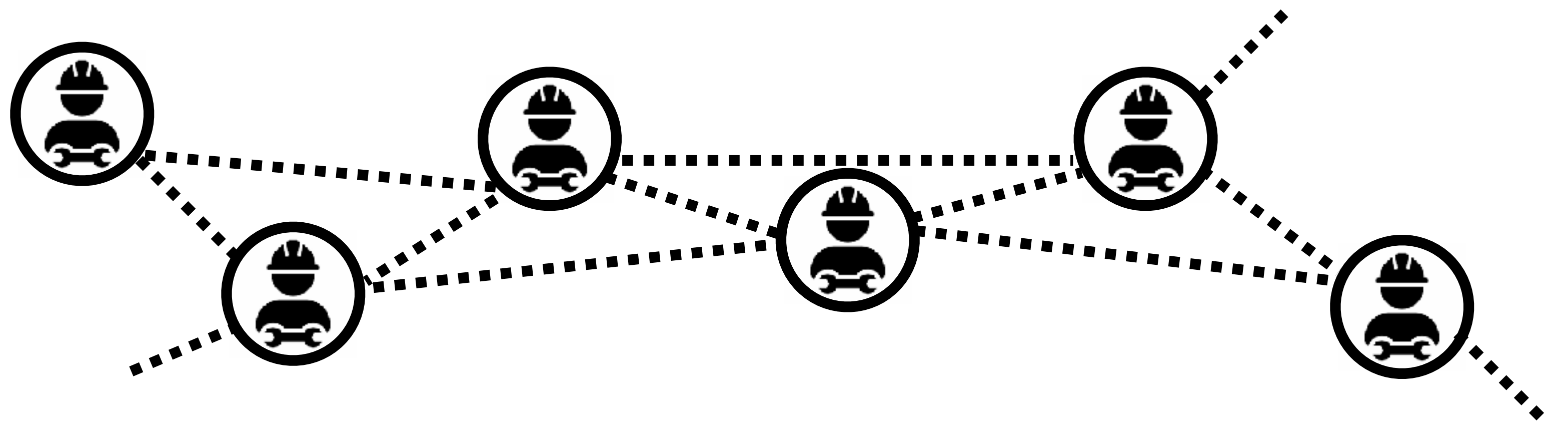
See book chapter 1 for details

Consensus in Blockchain networks

Who can be a validator?

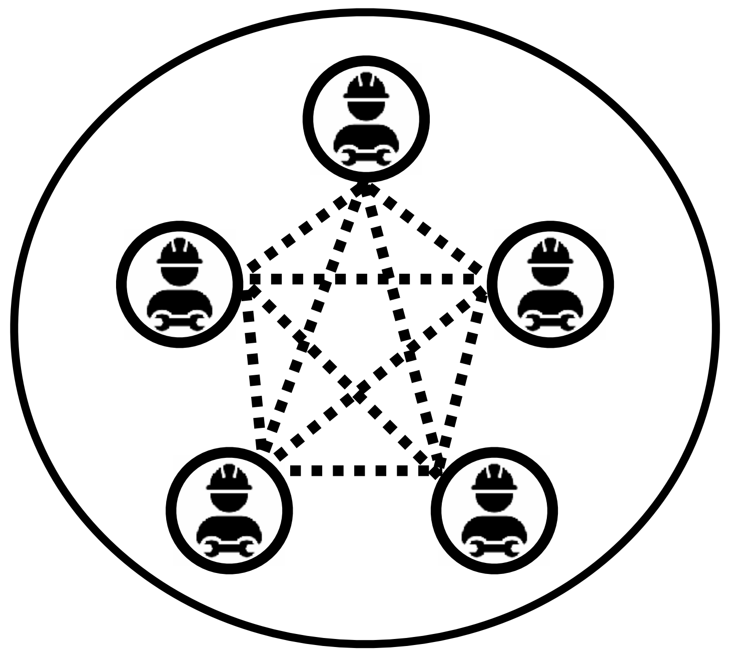
- In **permissionless** blockchains: anyone can join the network to become a transaction validator. No need to ask for permission to anyone. Group membership is ***open***.

- In **permissioned** blockchains: must receive *permission* from a coordinator or from existing validators in order to become a transaction validator. Group membership is ***closed***.



Open

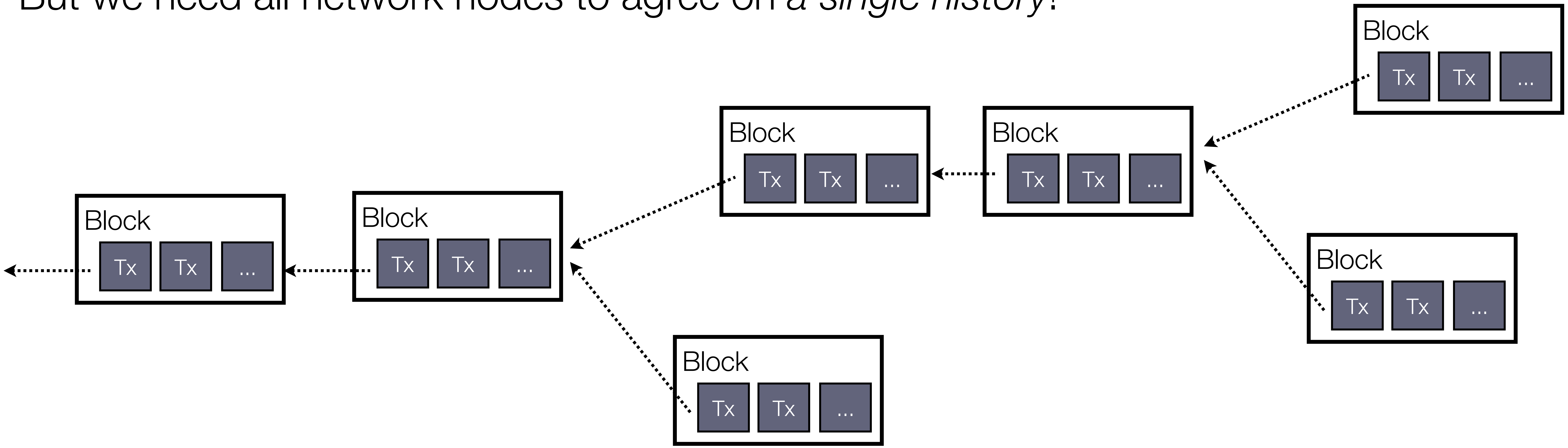
VS



Closed

Problem: diverging histories

- In an open system, if *anyone* can easily produce a valid block and add it directly to the ledger, there is little hope that the network will end up agreeing on a *single* ledger
- More likely, we would end up with a quickly growing *tree* of blocks
- But we need all network nodes to agree on *a single history*!



How to get consensus: organize a vote?

- We can let the network **vote** to **elect a single validator node** to propose the next block
- Ideally the proposer node is chosen **randomly** to avoid any bias in the election process
- But how to organize a vote in an open and permissionless network?
  - 1. We don't even have a fixed list of nodes to organize a voting poll
  - 2. Even if we would have a list of nodes, how to assign **voting rights** to each one?
- One IP address = one vote? Problem: attacker may control multiple IP addresses
- This is known as a **sybil attack**. The same problem holds for any other type of "identity" that is cheap to create (e.g. public keys)

How to get consensus: organize a lottery!

- To elect a node from an open group of participants, organize a **lottery**: each node “buys” tickets, whoever can “prove” they have the lucky ticket is the winner (and so gets to propose the next block)

- The lottery should have the following properties:

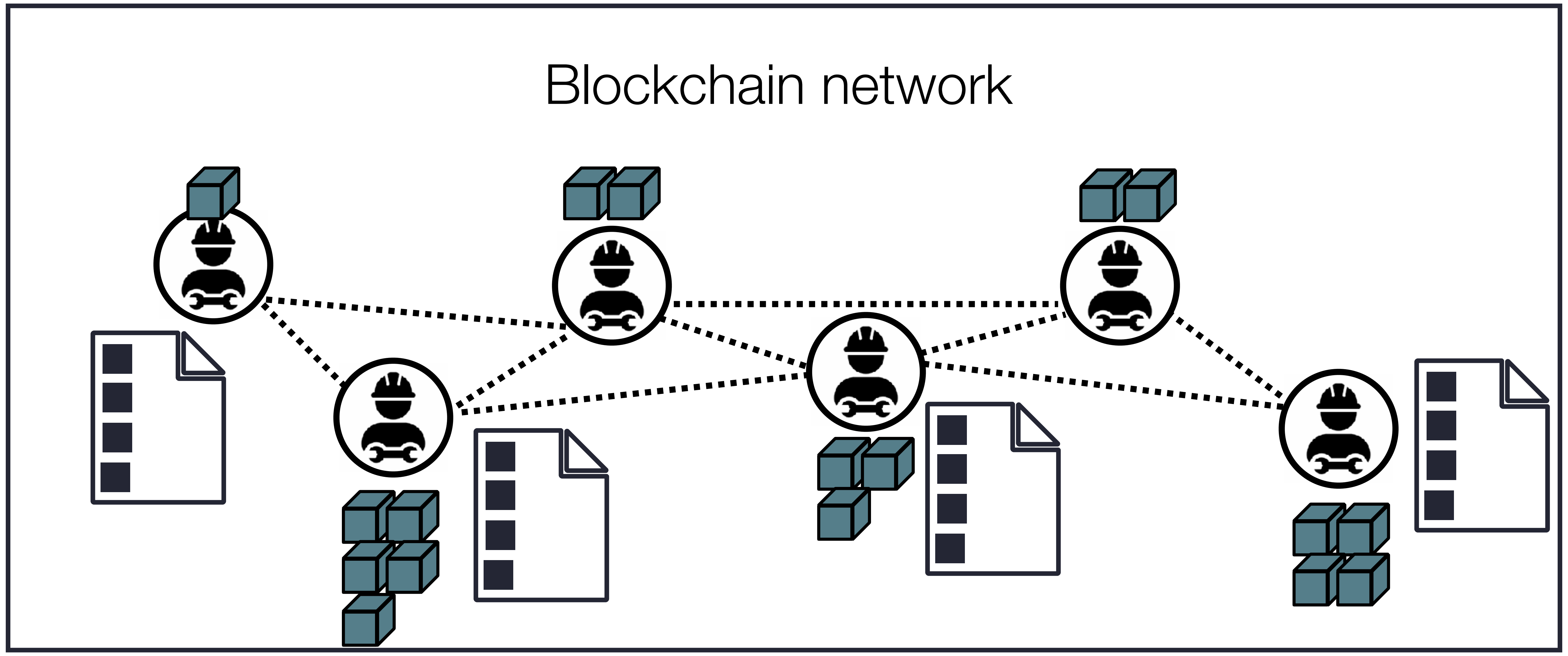
  - **Fair** - node election should be distributed across the broadest possible population of participants (i.e. “everyone can buy a ticket”)

  - **Proportional** - The cost of controlling the election process should be proportional to the value gained from it (i.e. “the more tickets bought, the higher the chance of winning”)

  - **Verifiable** - It should be relatively simple for all participants to verify that the winning node was legitimately selected (i.e. “everyone can verify whether the winning ticket is indeed a valid ticket”)

Lottery-based consensus in permissionless blockchains (“proof-of-X”)

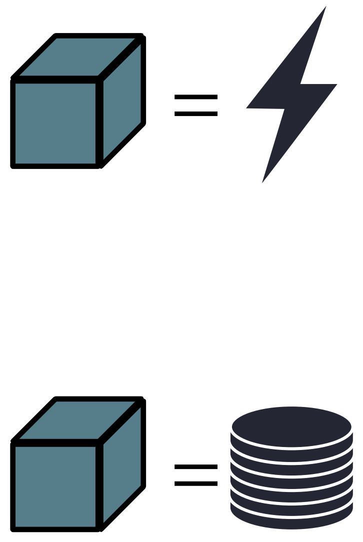
- Validators enter the lottery by proving ownership of some digital or physically scarce resource
- Different blockchain networks use different kinds of resources



Example lottery-based consenus protocols:

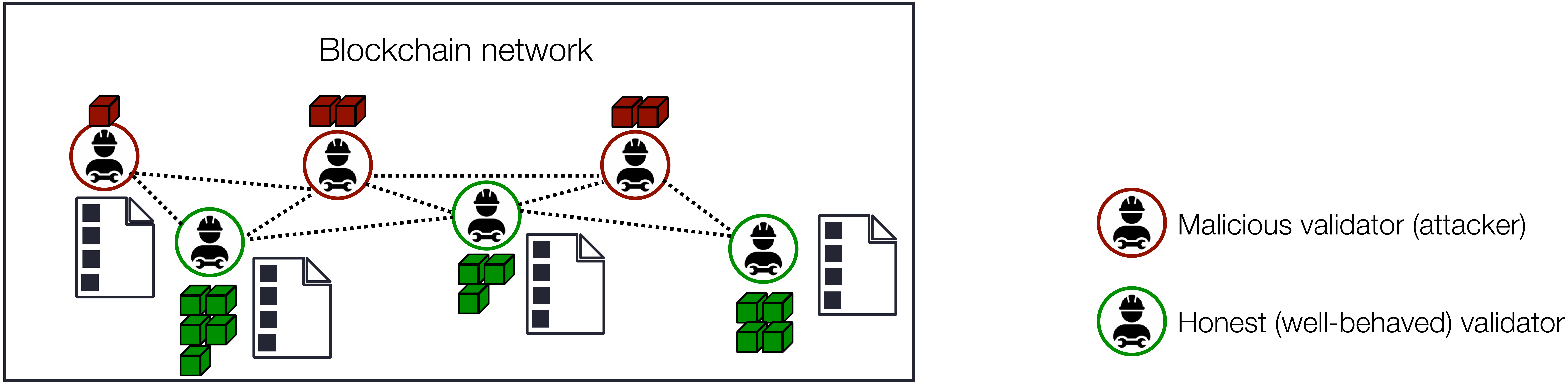
“Proof-of-work”
(vote with compute power)

“Proof-of-stake”
(vote with tokens)



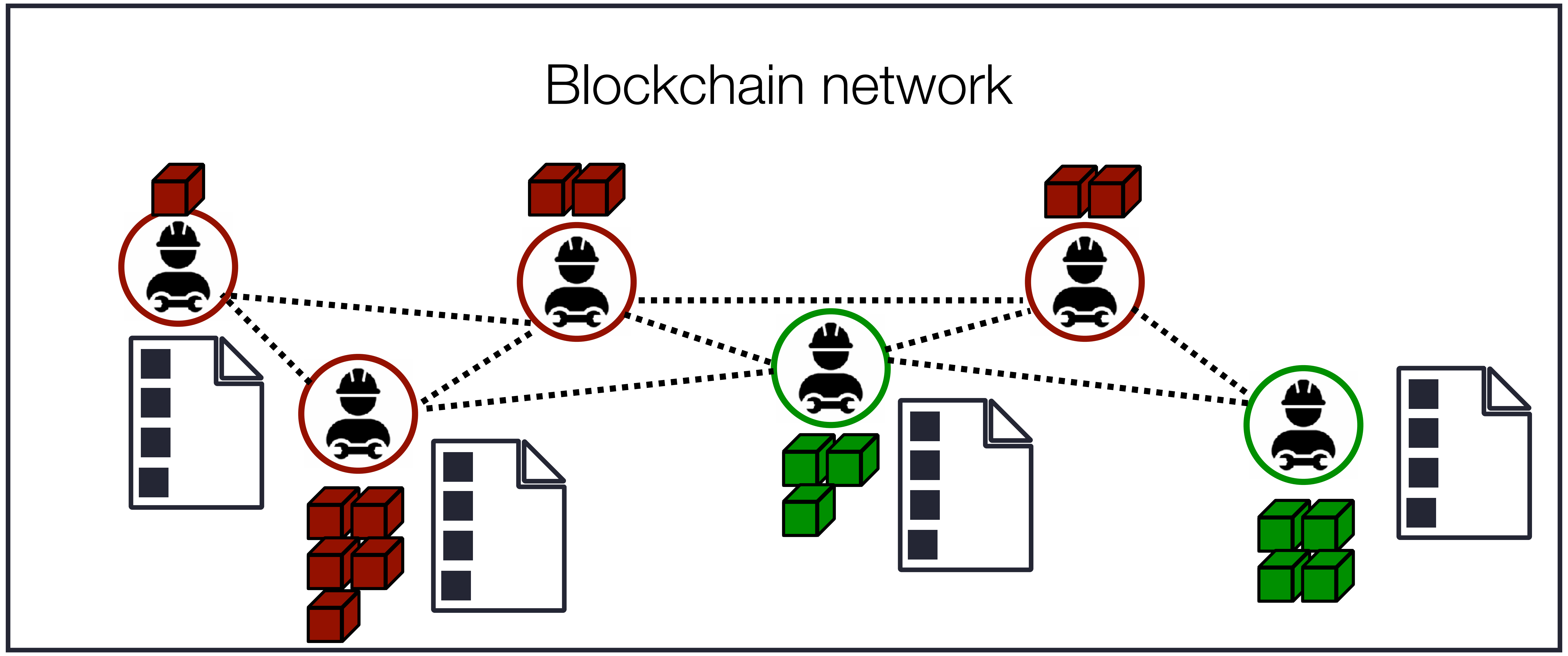
Lottery-based consensus in permissionless blockchains (“proof-of-X”)

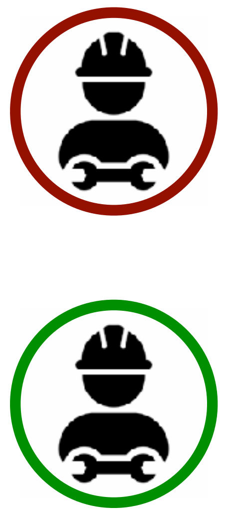
- The integrity of the blockchain is guaranteed as long as a **majority** of the network, **weighted** by their resource ownership, is controlled by well-behaved validators



Attacking a permissionless blockchain network: “51% attack”

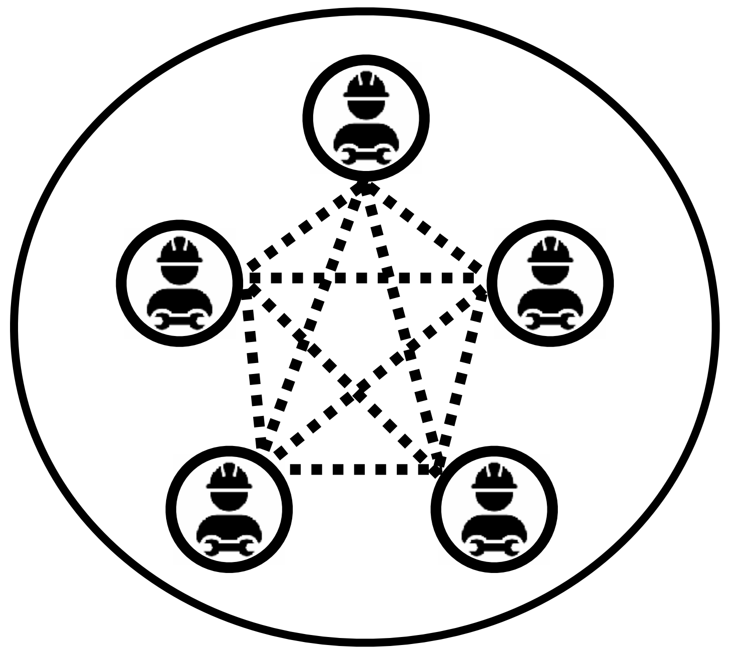
- If an attacker (or group of attackers) **controls >50% of the scarce resources**, they effectively control the production of new blocks.
- While such an attacker cannot create “fake” signed transactions (i.e. steal tokens), they can reject (**censor**) any number of transactions and can approve transactions that **double-spend** their own tokens by “forking” the blockchain and “rewriting” block history.





Consensus in Permissioned Blockchain networks

- Recall: permissioned blockchain networks **limit a priori** who can join the network to become a validator
- The group of validators is closed. This **avoids sybil attacks**.
- No need to use “**lottery**”-based consensus
- Instead, can use standard “**voting**”-based consensus algorithms
  - Crash Fault Tolerant (CFT) consensus algorithms (e.g. Paxos, Raft, …)
    - Can tolerate **< 1/2** failing validators, but **0** malicious validators!
  - Byzantine Fault Tolerant (BFT) consensus algorithms (e.g. PBFT, …)
    - Can tolerante **< 1/3** failing validators, **< 1/3** malicious (byzantine) validators

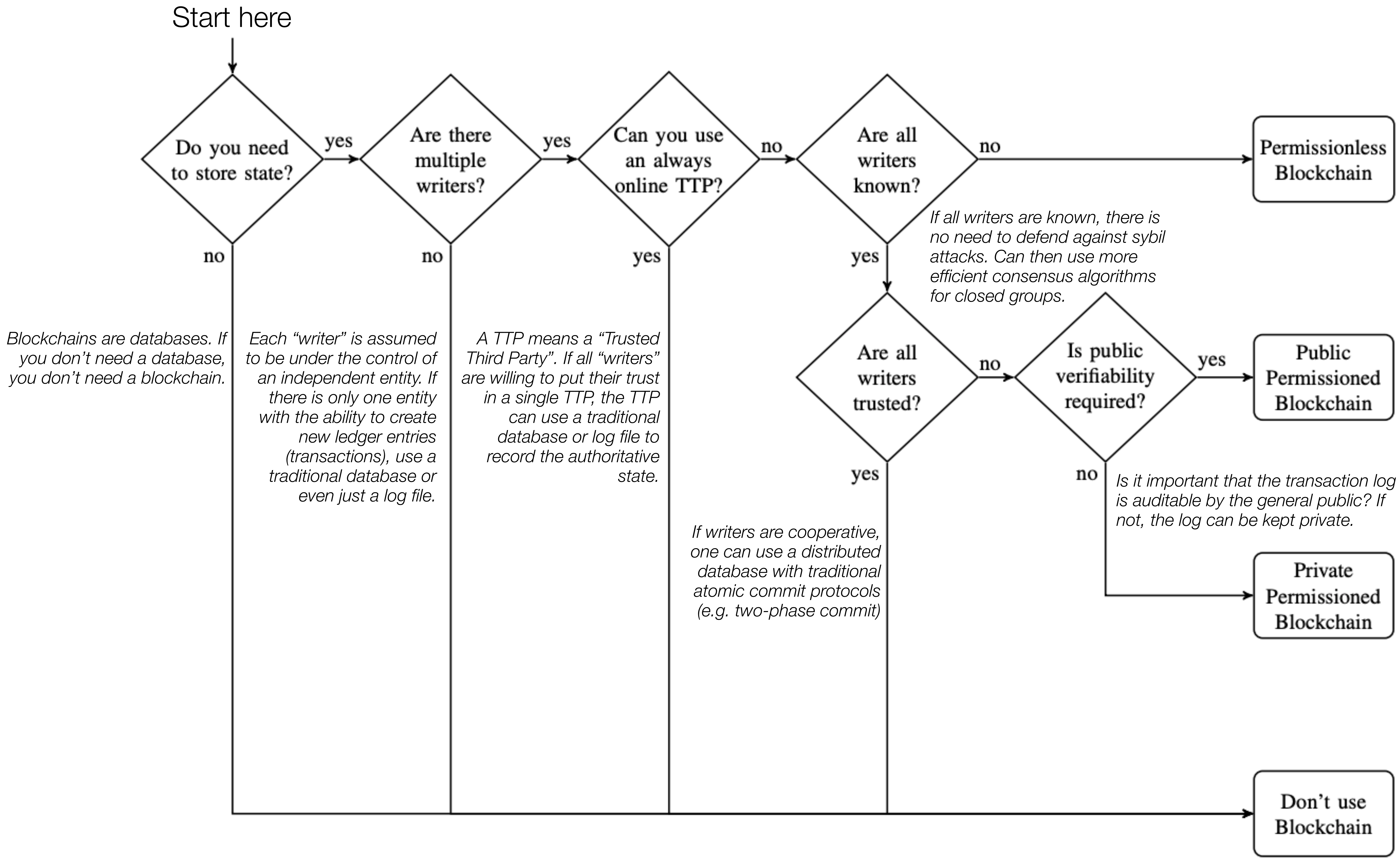


KU LEUVEN DistriNet

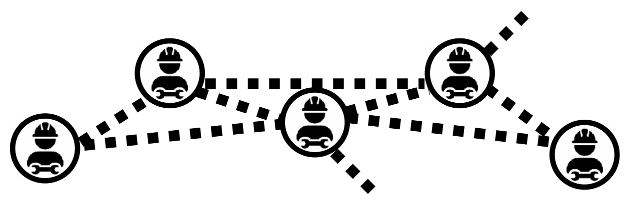
Permissioned vs Permissionless Blockchains

	Permissionless	Permissioned
Network peers	Are fully **anonymous** and **untrusted**	May or may not be anonymous. May have some level of **trust** based on external (business) incentives.
Consensus achieved via	**Lottery-based** algorithms, based on proof of owning some scarce resource (e.g. Proof-of-Work, Proof-of-Stake)	**Voting-based** algorithms, such as Byzantine Fault-tolerant (BFT) consensus algorithms (e.g. PBFT)
Peer membership	**Open** (anyone can join, no need to ask “permission” to join)	**Closed** (an administrator manages membership, or pre-existing members vote to update the membership list)
Energy-efficiency	Very **low** for Proof-of-Work **High** for Proof-of-Stake	**High** (similar to a standard replicated databases)
Transaction rate	**Low** (3-4 tx/sec for Bitcoin, 15-20 tx/sec for Ethereum). Generally: the larger the consensus group, the lower the TPS	**High** (10,000 or more TPS) (TPS = transactions per second)
Transaction finality	**Slow**. E.g. in Bitcoin transactions are considered “final” after 6 blocks, and each block takes ~10 minutes to produce)	**Fast**. Block **production times** on the order of a few **seconds**, 1 block confirmation is often sufficient.
Security (51% attacks)	Scales to **large networks** of *O*(1000s) nodes making it very **expensive** for an attacker to disrupt a majority of peers.	Deployed with *O*(10-100) nodes making it **more feasible** (but still difficult) for an attacker to disrupt a majority of peers.

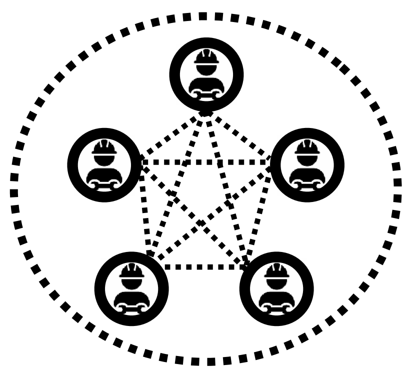
When (not) to use a blockchain?



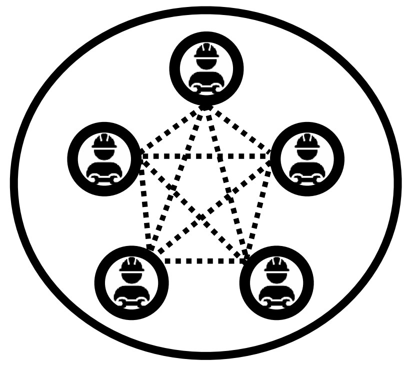
***Example**: public blockchain networks such as the Bitcoin or Ethereum network. Example use case: peer-to-peer payments, decentralized trading, creating provably-scarce collectible digital tokens.*



***Example**: a public blockchain hosted by a consortium of organizations, where only members of the consortium can write to the chain, but everyone can read the data on the chain. Example use case: companies issuing loyalty points or frequent flyer miles to customers.*



***Example**: a blockchain that is privately hosted by a consortium of companies, where only members of the consortium can write to the chain and read from the chain. Example use case: tracking the production and shipment of parts or goods across a manufacturing or logistics supply chain.*



(Source: K. Wüst and A. Gervais, "Do you Need a Blockchain?", CVCBT 2018)

KU LEUVEN DistriNet

Decentralized systems and Blockchain networks: summary

- Centralised vs **decentralized** distributed systems
- Why is a blockchain needed? Example: electronic cash and **the double spending problem**
- How are **transactions** processed in a blockchain network?
- How is **cryptography** used to securely record transactions on a blockchain?
- How is consensus achieved in a blockchain network in the face of **sybil attacks**?
- The difference between **permissioned** and **permissionless** blockchain networks
- Next lecture: a case study of the **Ethereum** blockchain network