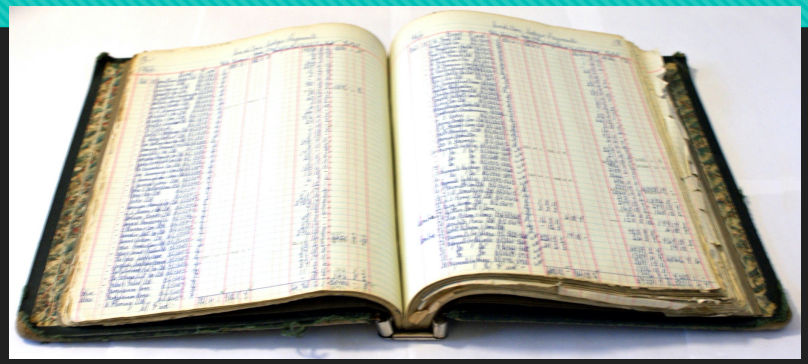
Blockchain as a Distributed Ledger (DLT)

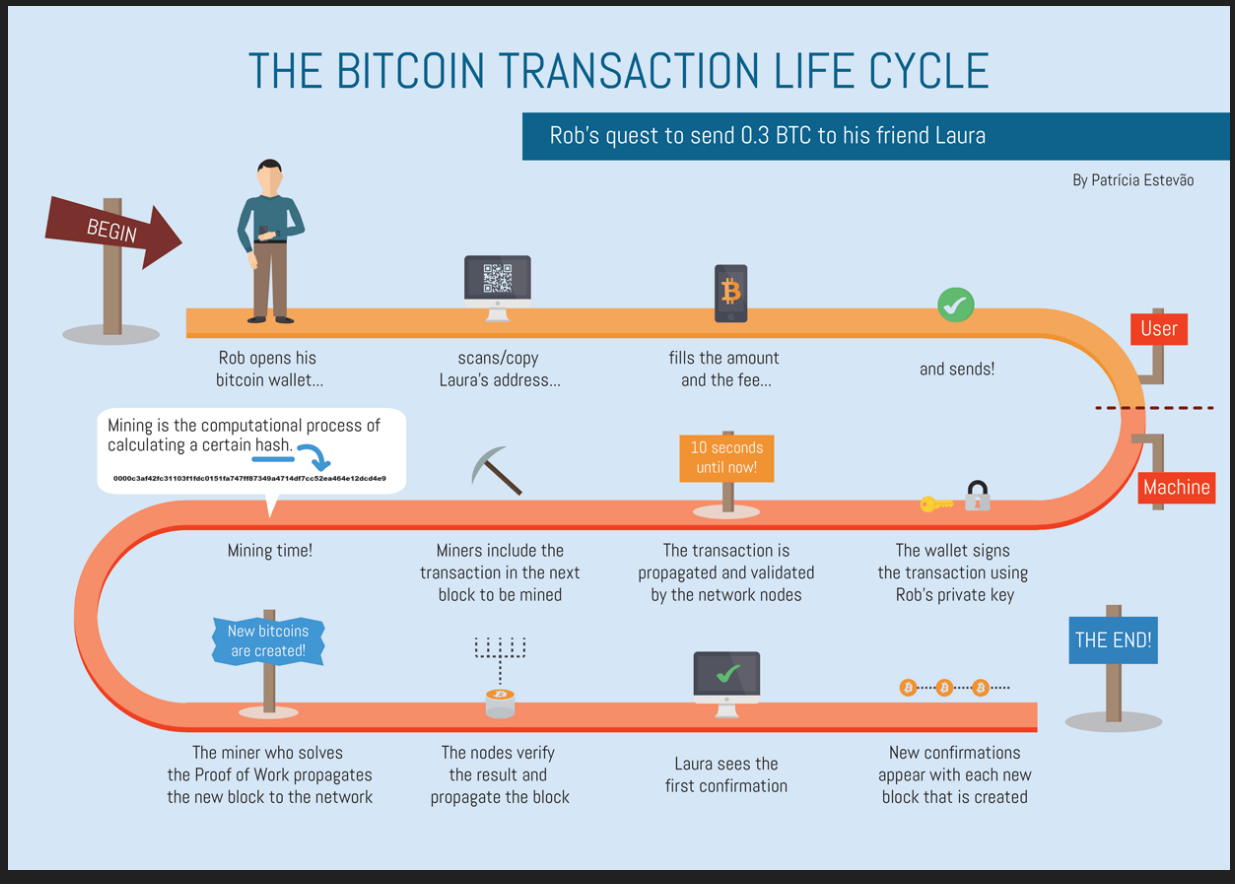


- Digital ledger = records history of all past transactions
  - i.e., the blockchain constitutes a transaction log / database
  - Transparently accessible to anyone
  - Can be shared on all markets
- Transactions between participants
  - Asset transfers, and most notably financial exchanges
  - Cryptocurrency managed through the blockchain
    - Bitcoin as an example
- Distributed and shared
  - Replicated = not a single book, but many replicas of the log
  - Produced collaboratively
  - Same as multiple accountants writing the same log into multiple books

Replicated and consistent data storage

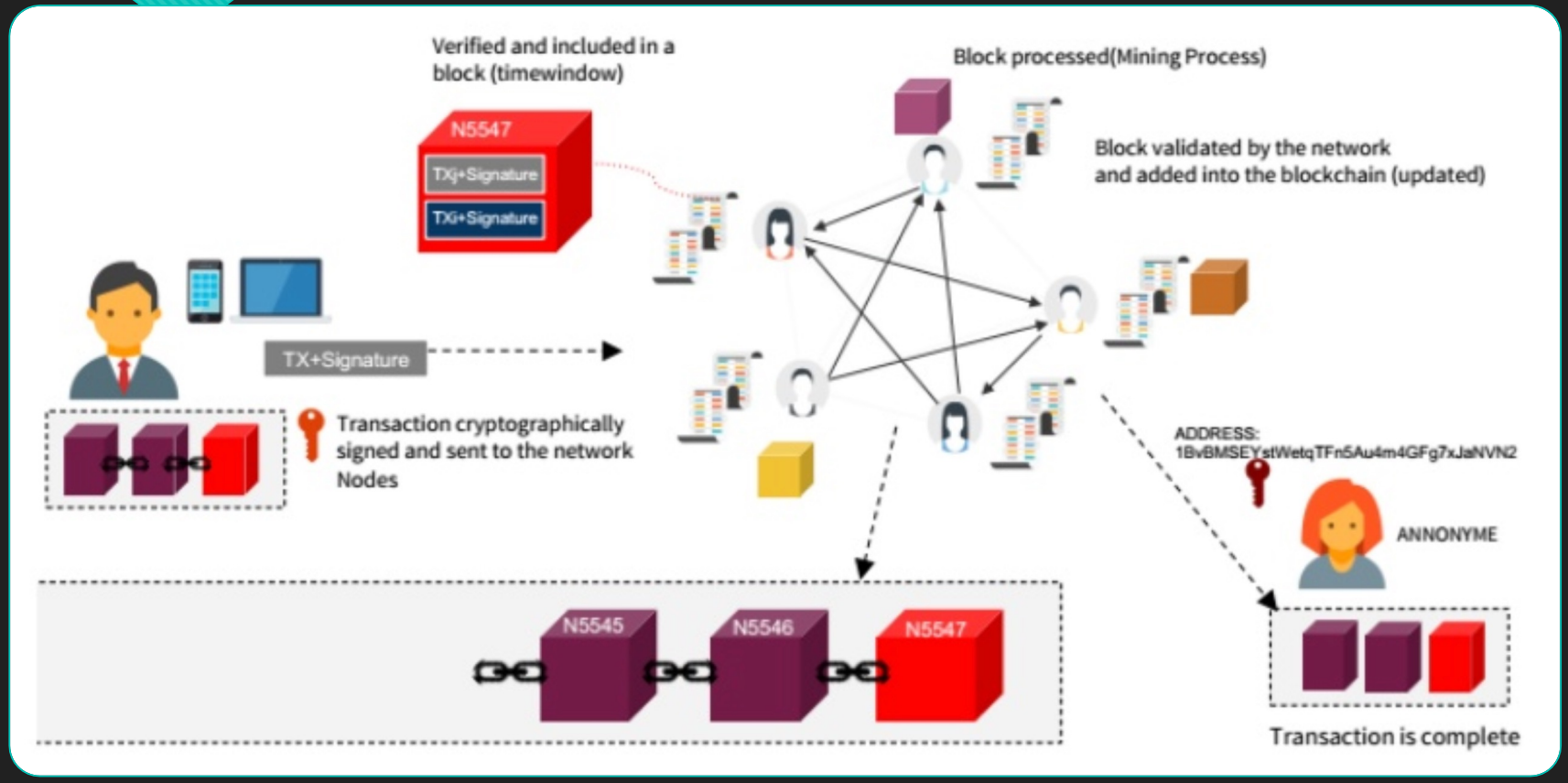
- Organization of nodes (P2P style)
  - Each node has a copy of the whole blockchain
  - Each node has a registry of all nodes in the network
- Data replication: no Single Point of Failure (SPOF)
  - All nodes are notified about state change (transition broadcast)
  - Each node records it in its replica of the blockchain
  - Makes it possible to achieve high availability (node crash, DoS)
  - Finally completed blocks are broadcasted too (after their integrity is secured)
- Consistency handling and conflict resolution
  - Replication requires dealing with inconsistent broadcasts (lost messages)
  - Must also deal with attempts to inject bogus records
  - Validation relies on P2P consensus, achieved through cryptographic proofs (PoW, PoS, etc.)

Transactions – the process - user perspective



https://www.weusecoins.com/images/bitcoin-transaction-life-cycle-high-resolution.png

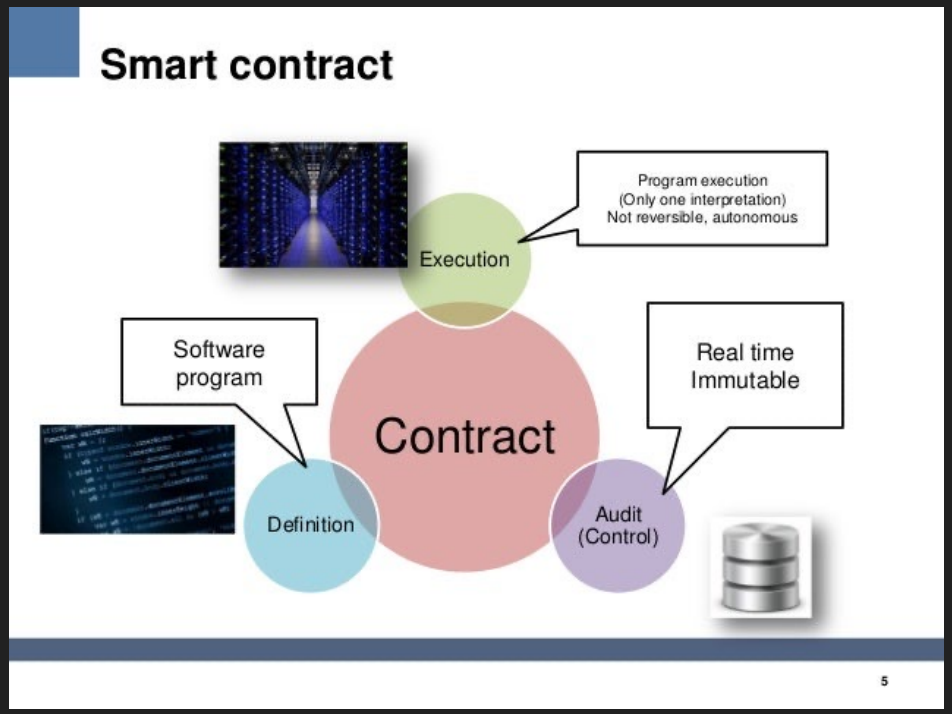
Transactions: Parties involved in validation and logging



Blockchain as a decentralized middleman

- Smart contracts: agreement between multiple parties stored into the blockchain
  - Introduced with the Ethereum blockchain, also available now over different blockchains (including Bitcoin)
  - small decentralized programs embedded into the ledger(Autonomous agents reborn!)
- Properties
  - Contracts are stored into the blockchain and cannot be altered or suppressed
  - Its automatic execution is enforced by the consensus protocol and can be verified by anybody
  - Relies on cryptocurrency for contract resolution (not necessarily within the blockchain, e.g., Hyperledger)
  - No permission required to deploy
- Application examples
  - IT processes: Decentralized DNS (e.g., http://namecoin.info)
  - Trustless contracts and their negotiation without intermediaries: loans, property records, capital markets, …
  - Trustless payment for various services: logistics, energy grids, transportation systems, …
  - Crowd investing in virtual organizations (ICOs and DAOs)
  - …

From traditional to smart contracts

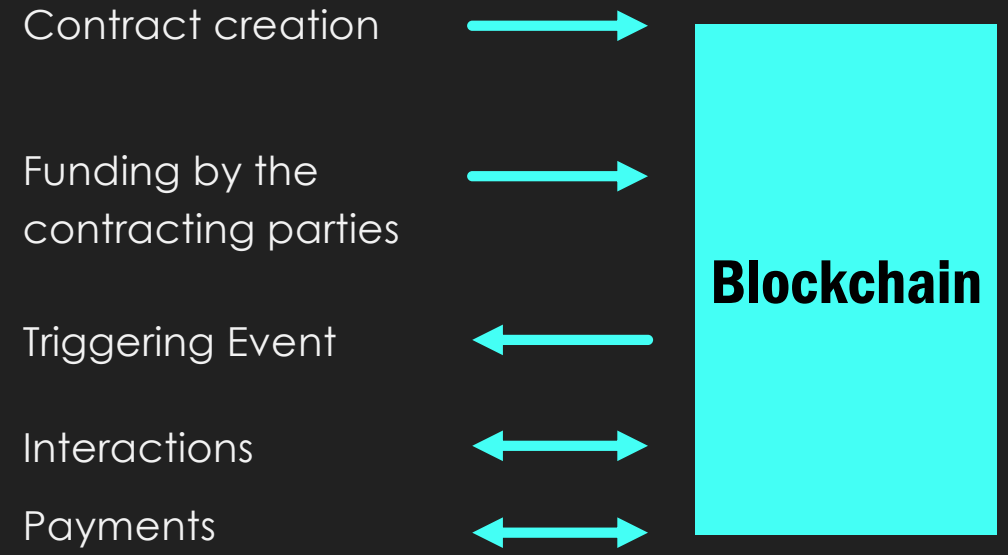


https://image.slidesharecdn.com/smart-contracts-150925125324-lva1-app6892/95/smart-contracts-4-638.jpg?cb=1443185644

https://image.slidesharecdn.com/smart-contracts-150925125324-lva1-app6892/95/smart-contracts-5-638.jpg?cb=1443185644

Smart contract

- Transaction = arbitrary code
- Logic responds to events on blockchain
- May transfer assets in response
- Can implement
  - Auction
  - Investment decision
  - Money transfer
  - Vote
  - …



Smart contract example

- Code in Solidity (Ethereum language)
  - This function sends an amount of coins to the receiver's address from the address calling the function. For example, if Bob calls this function with Alice's address and the amount is 1000, then 1000 coins will be transferred from Bob's account to Alice's account.

```
pragma solidity ^0.4.0;

contract Coin {
 // The keyword "public" makes those variables
 // readable from outside.
 address public minter;
 mapping (address => uint) public balances;

 // Events allow light clients to react on
 // changes efficiently.
 event Sent(address from, address to, uint amount);

 // This is the constructor whose code is
 // run only when the contract is created.
 function Coin() {
 minter = msg.sender;
 }
 function mint(address receiver, uint amount) {

 if (msg.sender != minter) return;
 balances[receiver] += amount;
 }

 function send(address receiver, uint amount) {

 if (balances[msg.sender] < amount) return;
 balances[msg.sender] -= amount;
 balances[receiver] += amount;
 Sent(msg.sender, receiver, amount);
 }
}
```

Crowd investing: DAOs

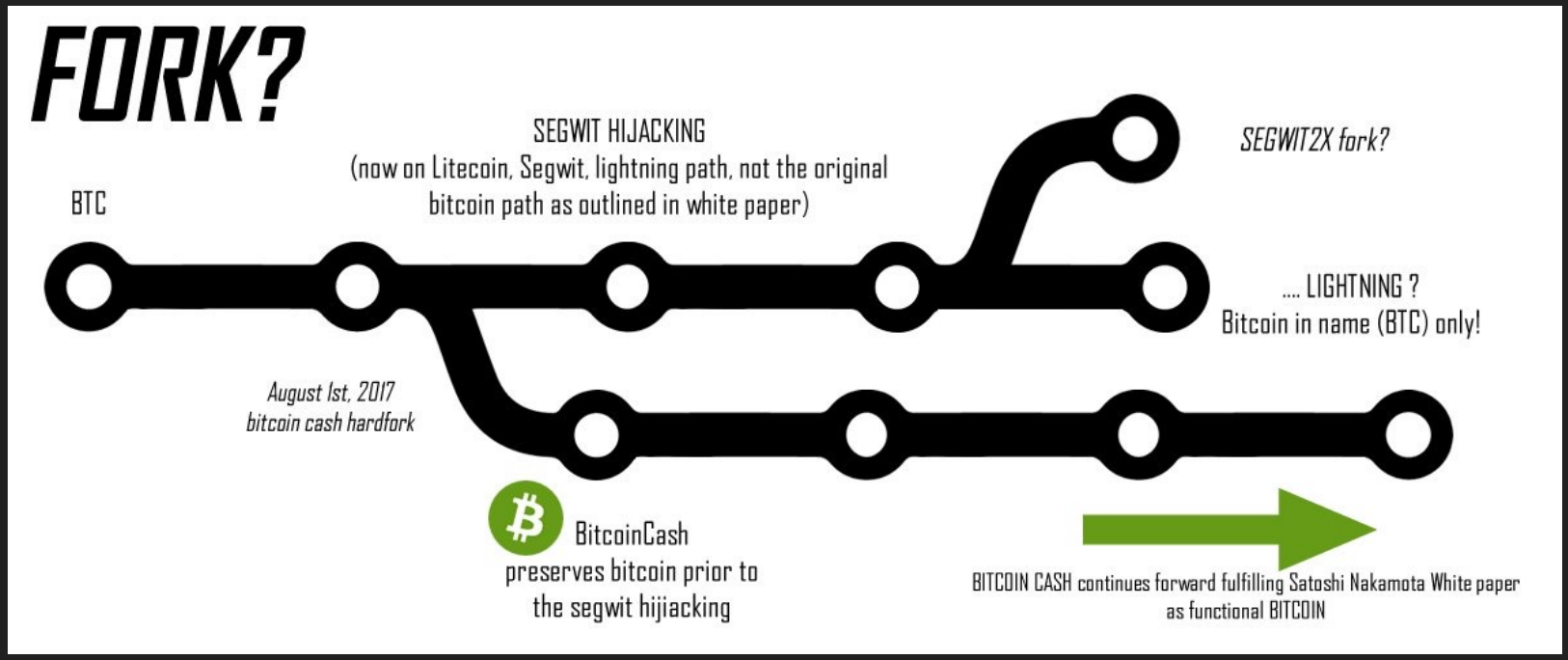
- Decentralized Autonomous Organizations (DAO)
  - CEO-less organization / project
  - cryptographically guaranteed “democratic” fund management through decentralization
  - stakeholders can vote on adding new rules, changing the rules, ousting a member, make investment decisions, or hire contractors for making products
- DAO management structures only need the blockchain to be deployed
- Major problem: once started, one can’t stop a DAO
  - Even if its code contains a bug

Crowd investing: Initial Coin Offerings (ICO)

- DAOs are funded by Initial Coin Offering (ICO) or "token sales"
  - Fundraising mechanism
  - Participants receive digital tokens used later to run the DAO
  - The investment in tokens will generally correspond to the return (financial or else)
- Typically used by startups to fund their activities (so far)
  - Tezos: 232 million dollars (July 2017)
  - Filecoin: 200 million dollars (September 2017)
  - Brave: 35 million dollars in less than 30 seconds (May 2017)
  - Cosmos: 16 million dollars in less than 30 minutes (April 2017)
- Opening venture-capital investment to anybody
  - Yet no legal basis contrary to traditional investment vehicles
  - And also opens the door to scams!
  - Advertisement has been banned from many social networks

Blockchain forks

- Bitcoin example



NFTs – Non-Fungible Tokens

- Token which certifies the authenticity and/or prove the ownership of a digital asset
- Its integrity and uniqueness are protected through blockchain transactions
- Can be sold or traded.
- Many recent developments: digital art, games, music, film
  - The work of art is generally not the NFT, and can be copied
  - Legal foundations of NFTs are shaky (in comparison with copyright)

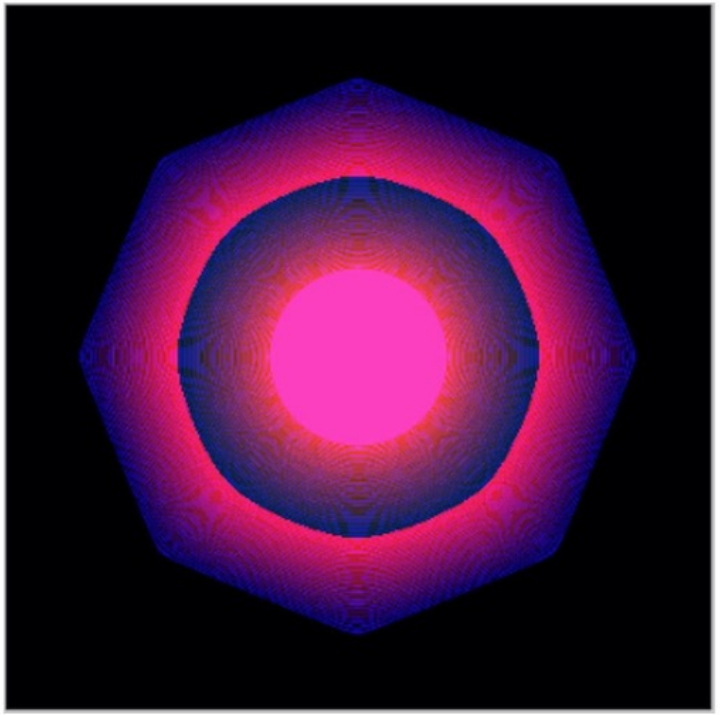
NFTs

- May 2014: first NFT crafted (named « Quantum »), auctioned in 2021
- OpenSea platform (opensea.io) is the most important for trading art-related NFTs

Sotheby's Is Selling the First NFT Ever Minted —and Bidding Starts at $100

The offering is part of a curated sale of NFTs called "Natively Digital."

Sarah Cascone, May 7, 2021



Kevin McCoy, *Quantum* (2014). The first NFT ever minted is now being offered at Sotheby's New York. Courtesy of Sotheby's.

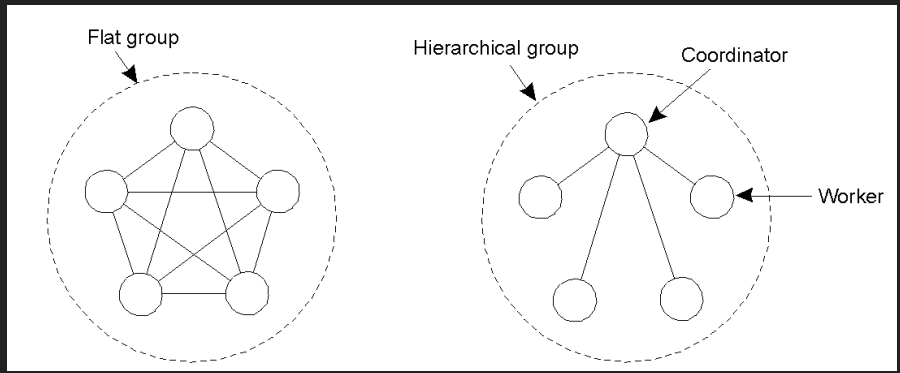
https://news.artnet.com/market/sothebys-is-hosting-its-first-curated-nft-sale-featuring-the-very-first-nft-ever-minted-1966003

Origins of the blockchain mechanisms

- **Consensus mechanisms:**
  - protocols for synchronizing distributed nodes and reaching agreement in a decentralized fashion despite Byzantine faults
  - e.g., State machine replication: fault-tolerant service by (actively) replicating servers and coordinating client interactions with server replicas (PBFT, Byzantine Paxos, Tangaroa)
- **Computational puzzles:**
  - slowing attackers through the execution of expensive proofs
  - e.g., Hashcash
- **Peer-to-Peer networks:**
  - distributed and dynamic infrastructure for supporting and load-balancing joins and leaves on a large scale, often supported through distributed data structures like distributed hash tables (DHTs) or Merkle DAGs, and possibly cooperation incentives
  - e.g., BitTorrent, Napster, Kazaa, Gnutella, CAN, Chord, Pastry, Tapestry, Kademlia
- **e-Cash:**
  - authenticating coins through blind signatures
  - e.g., CFD / e-cash

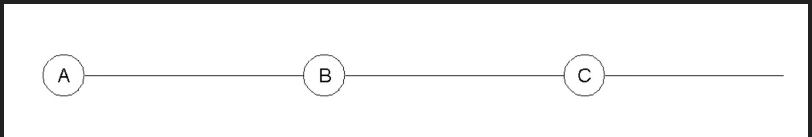
Groups: Achieving Process Resilience

- Objective: mask process failure by deploying redundant processes
  - Processes with similar functionalities form a group
  - Static or dynamic (join and leave primitives)
  - Flat groups
    - Totally decentralized decisions, no single point of failure
    - Delay in decision making and communication overhead
  - Hierarchical groups
    - Coordinator based: faster decisions
    - Coordinator failure requires additional mechanisms (leader election for instance)



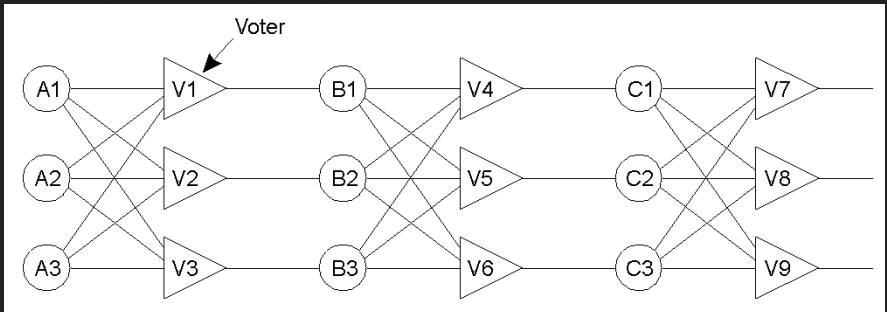
Groups: Replication Requirements

- *k fault tolerant* system = a system that can survive faults in k components and still meet its specifications
  - Fail-stop failures
    - K+1 processes are required at least, out of which 1 process would remain up
  - Byzantine failures
    - disobey protocol, send contradictory messages, collude with malicious processes …
    - 2k+1 processes are required at least
    - k+1 of these processes would still provide valid results
    - Fault-tolerant operation decided upon a vote among all processes
  - Both models assume requests are delivered in the same order to all servers
  - Example: Triple Modular Replication (TMR)



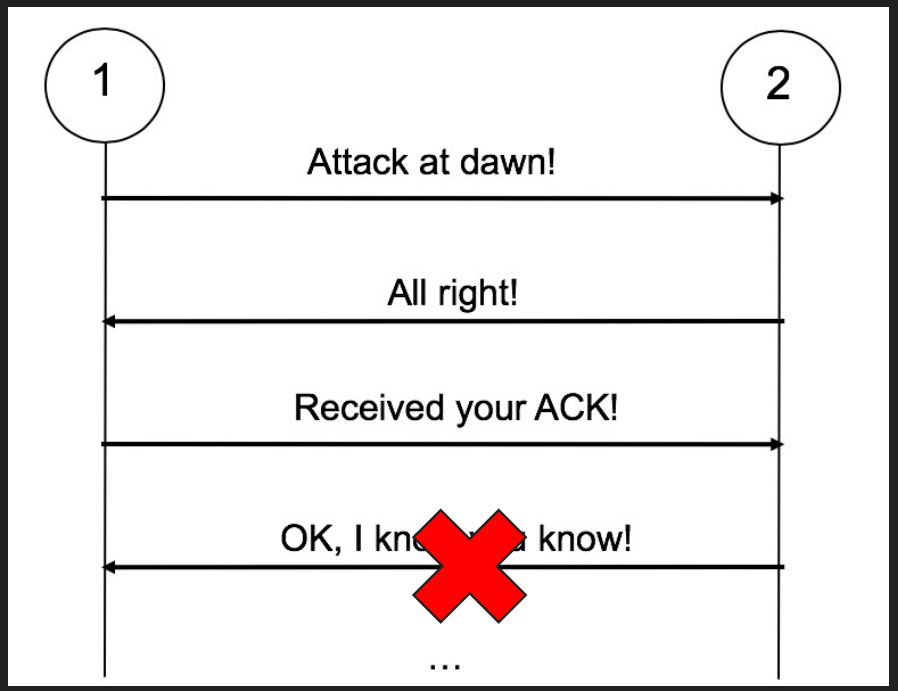
Groups: Replication Requirements

- *k fault tolerant* system = a system that can survive faults in k components and still meet its specifications
  - Fail-stop failures
    - K+1 processes are required at least, out of which 1 process would remain up
  - Byzantine failures
    - disobey protocol, send contradictory messages, collude with malicious processes …
    - 2k+1 processes are required at least
    - k+1 of these processes would still provide valid results
    - Fault-tolerant operation decided upon a vote among all processes
  - Both models assume requests are delivered in the same order to all servers
  - Example: Triple Modular Replication (TMR)



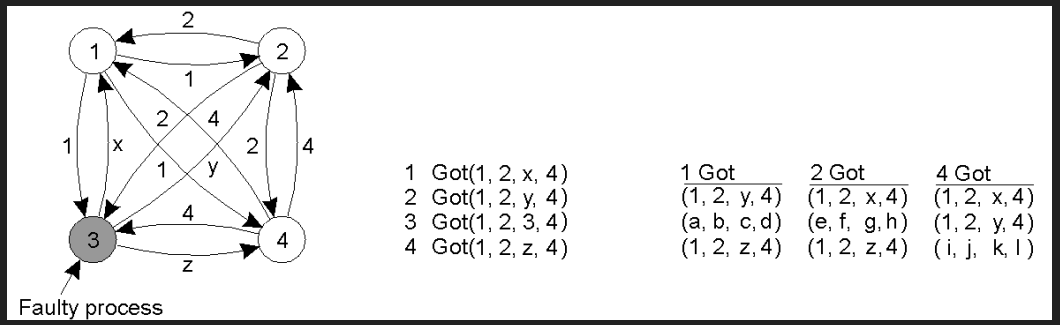
Groups: Agreements

- Byzantine *network* failures
  - The *two-army problem*
    - Generals must attack together to succeed!
  - Agreement cannot be reached in the face of unreliable communication
    - even with non-faulty processes!



Groups: Byzantine Agreement

- Byzantine *process* failures
  - The *Byzantine generals' problem*
    - Loyal generals and traitors
    - Instantaneous and perfect communications
    - Exchange information about troop strengths
  - Lamport et al. algorithm (1982)
    - k faulty processes
    - Agreement can be reached with at least 3k+1 processes (2k+1 working properly)
  - Example: 3 loyal generals and 1 traitor.
    - The generals announce their troop strengths and assemble vectors based on algorithm:

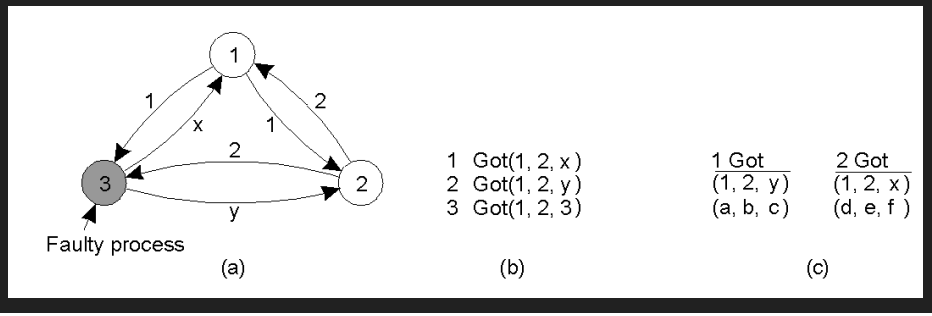


1 Got(1, 2, x, 4)
2 Got(1, 2, y, 4)
3 Got(1, 2, 3, 4)
4 Got(1, 2, z, 4)

1 Got	2 Got	4 Got
(1, 2, y, 4)	(1, 2, x, 4)	(1, 2, x, 4)
(a, b, c, d)	(e, f, g, h)	(1, 2, y, 4)
(1, 2, z, 4)	(1, 2, z, 4)	(i, j, k, l)

Groups: Byzantine Agreement

- Byzantine *process* failures
  - The *Byzantine generals' problem*
    - Loyal generals and traitors
    - Instantaneous and perfect communications
    - Exchange information about troop strengths
  - Lamport et al. algorithm (1982)
    - k faulty processes
    - Agreement can be reached with at least 3k+1 processes (2k+1 working properly)
  - Example: 2 loyal generals and 1 traitor.
    - Algorithm fails: agreement cannot be reached!



Consensus in Blockchains (1/2)

- Consensus protocol is the way to beat Byzantine failure
- neutral wrt. participants
- Different approaches to validation
  - Cryptographic protocol depends if permissionless or permissioned blockchain
  - Decentralized for permissionless blockchains (PoW, PoS, PoR, BA)
  - Based on authority proofs (PoA, DPoS) or consortium consensus (PBFT) for permissioned blockchains
- Some blockchains … aren't really blockchains!
  - For instance, IOTA is a distributed ledger (DLT) without blocks whose transactions are chained through a directed acyclic graph (DAG) with multiple ramifications
  - Validation relies on the DAG properties

Consensus in Blockchains (2/2)

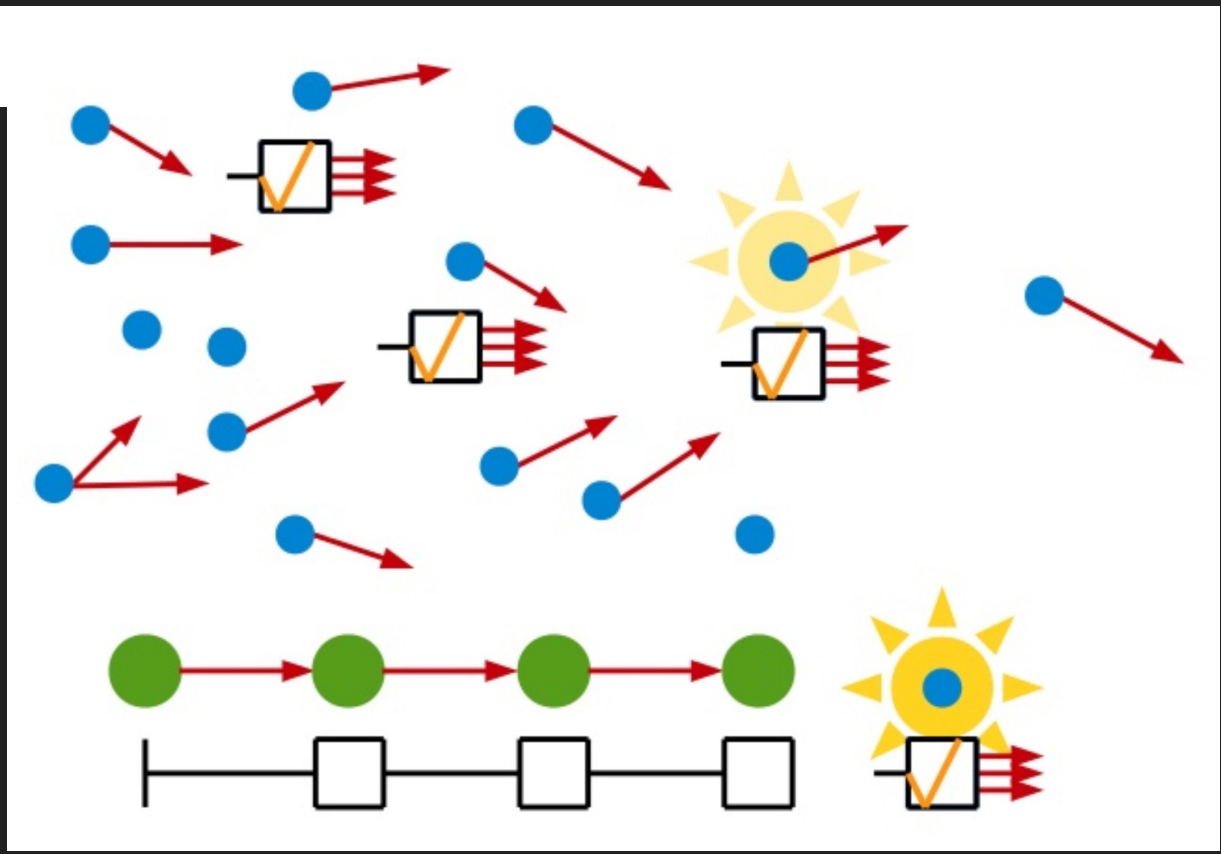
- PoW: Proof of Work (e.g., Bitcoin, Ethereum up to v3)
  - Solution of a cryptographic puzzle
- PoS: Proof of Stake (Ethereum from v4) or Proof of Possession
  - Nodes that prove their possession of a stake (coins) used as a guarantee then become eligible as blockchain validators – less energy consumption than PoW
- DPoS: Delegated Proof of Stake
  - An eligible PoS node delegates its authority to other nodes
- PoR: Proof of Retrievability or Proof of Storage
  - A transaction can be validated based on the proof that a node reserves a certain disk space to storage services, which can be checked by dedicated verifiers through random checks.
- PoA: Proof of Authority
  - Blocks and transactions are only validated by approved nodes acting as blockchain admins
- BFT: Byzantine Fault Tolerance
  - A fixed number of servers is designated to decide on the consensus, with the risk of fake nodes popping up in order to get in control of the consensus protocol (Sybil attack).
- BA: Byzantine Agreement
  - An improvement over BFT in the sense that validators' contribution is weighted based on the number of coins they own, thereby reducing the risk of a Sybil attack

Proof of Work (PoW)

- Principle: one CPU = one vote
- Ensures there are computing resources invested (slows down attackers)
- Hash as PoW (hard to compute, easy to verify)
- Bitcoin validation: costly (100 GB of transactions as of beginning of 2017)
  - verify hashes and digital signatures
  - verify absence of double spending
- Ethereum: more complex and costly
  - need to re-run all the smart contract code

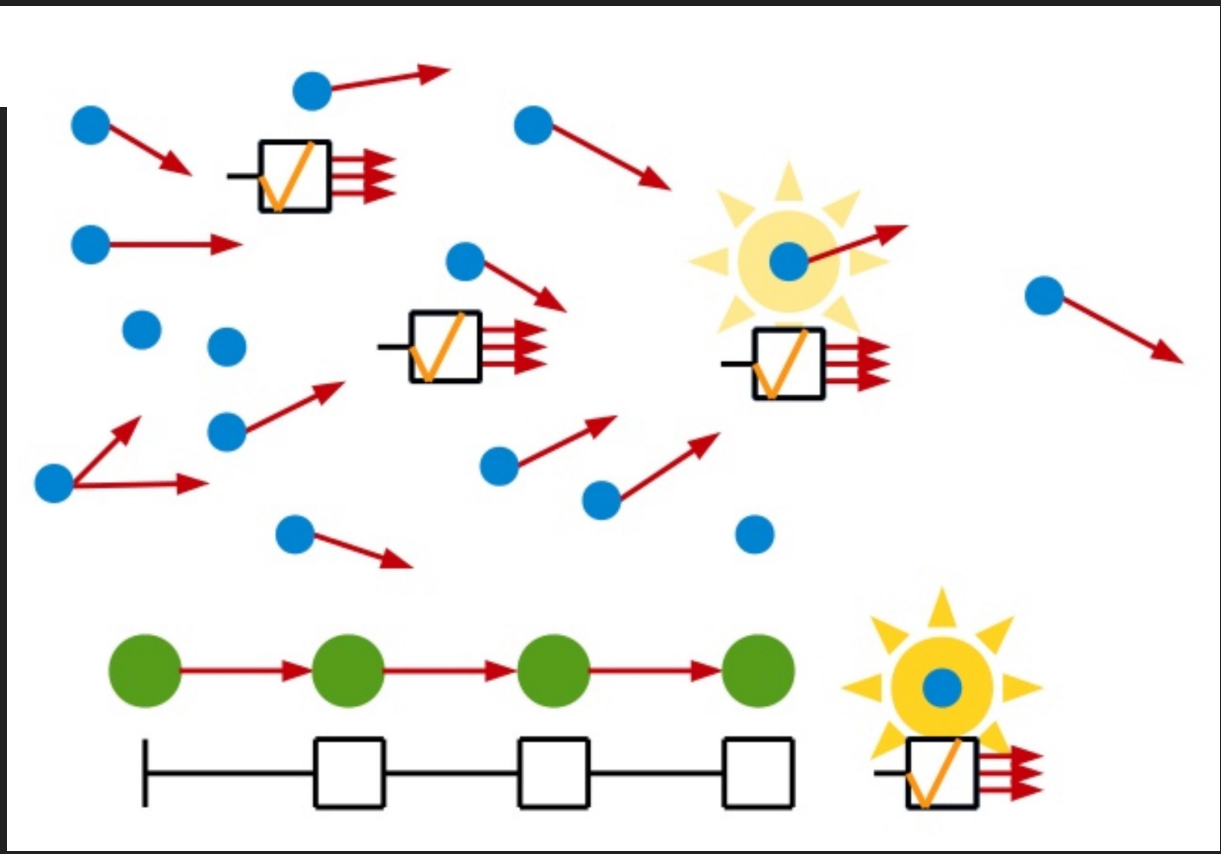
Bitcoin consensus (1/2)

- Bitcoin mining: lottery race
  - Solve a hard puzzle (1 succeeds every 10 min)
    - Winning miner: First come, first served
    - Winner's block is broadcasted
  - Winner obtains:
    - 6.25 newly created BTC reward for block creation
    - transaction fees from originators of transactions
  - Reward (=inflation) halved every 210000 blocks (about 4 years - previous halving: 2020, next expected: 2024)

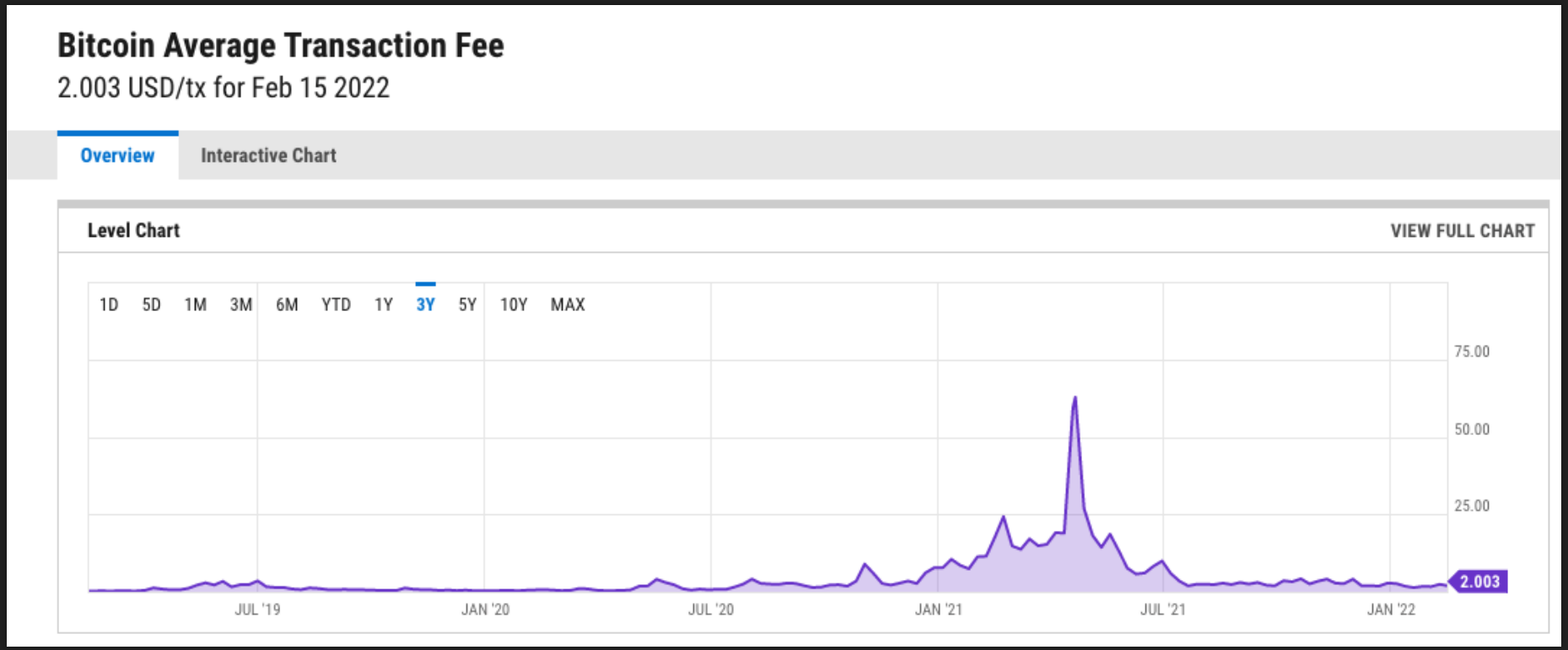


Bitcoin consensus (2/2)

- New block verification and validation
  - Verification means checking if a coin has been double spent
  - Verification also means checking hash
  - Earliest validated blockchain with most proofs ("longest") becomes the new reference for all nodes
  - Other blocks are temporarily stored by nodes, but miners have to choose a new main chain to mine their blocks (they « vote » with their computational power)
  - Transaction fee may motivate validators to select one transaction over another one



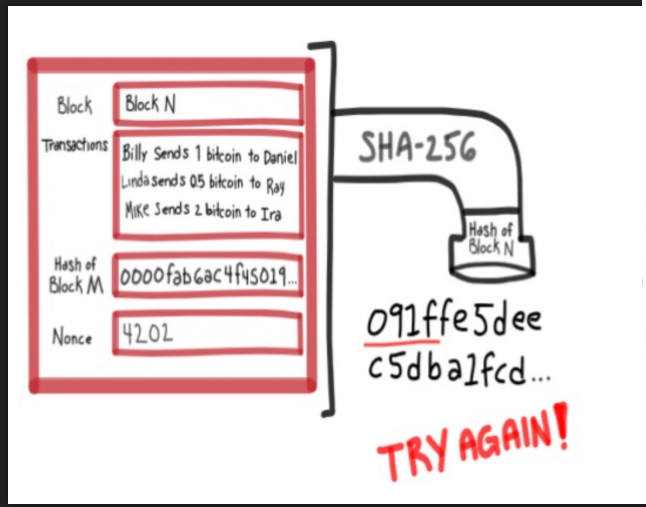
Transaction fees over time

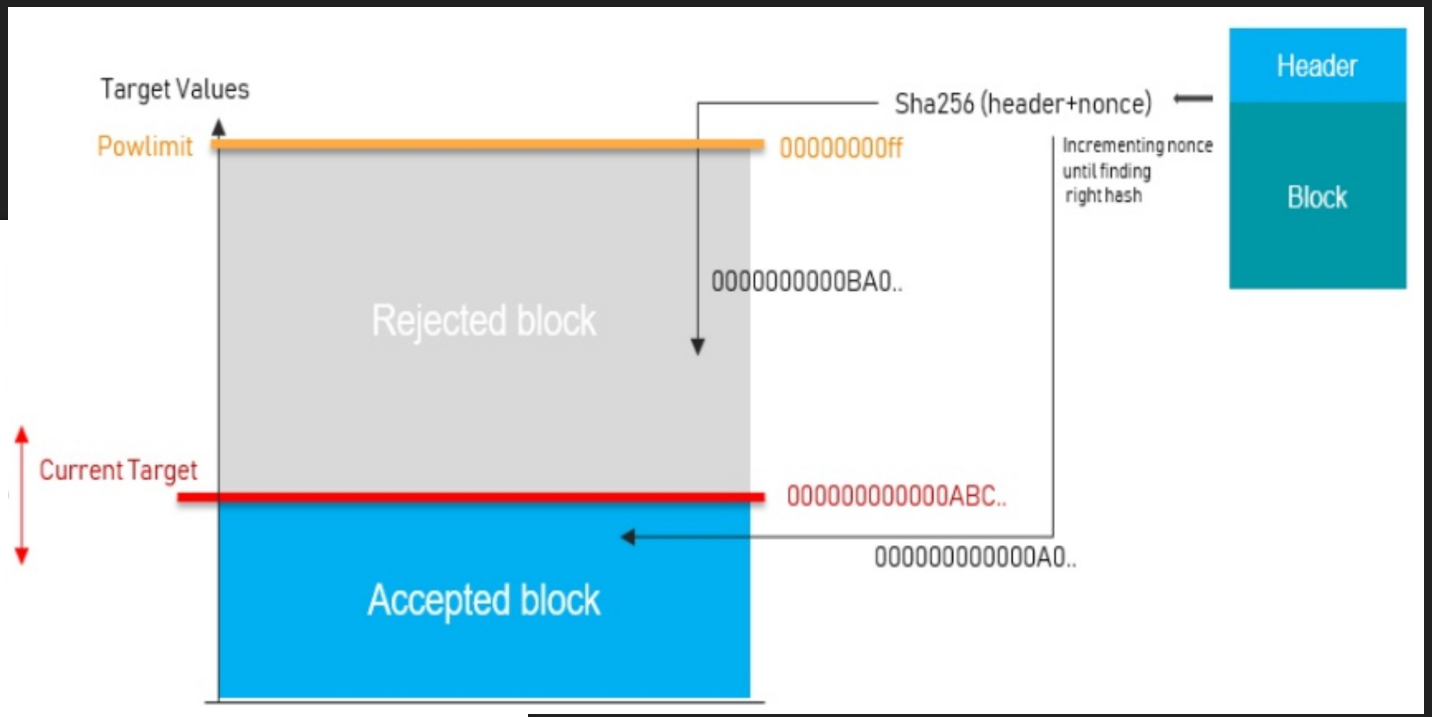


https://ycharts.com/indicators/bitcoin_average_transaction_fee

Bitcoin Mining: Hash based PoW

- Miner tries a nonce
- Hash computation: random value between 0 and 256-bit
- If hash < target value, miner wins, otherwise he must try with another nonce!



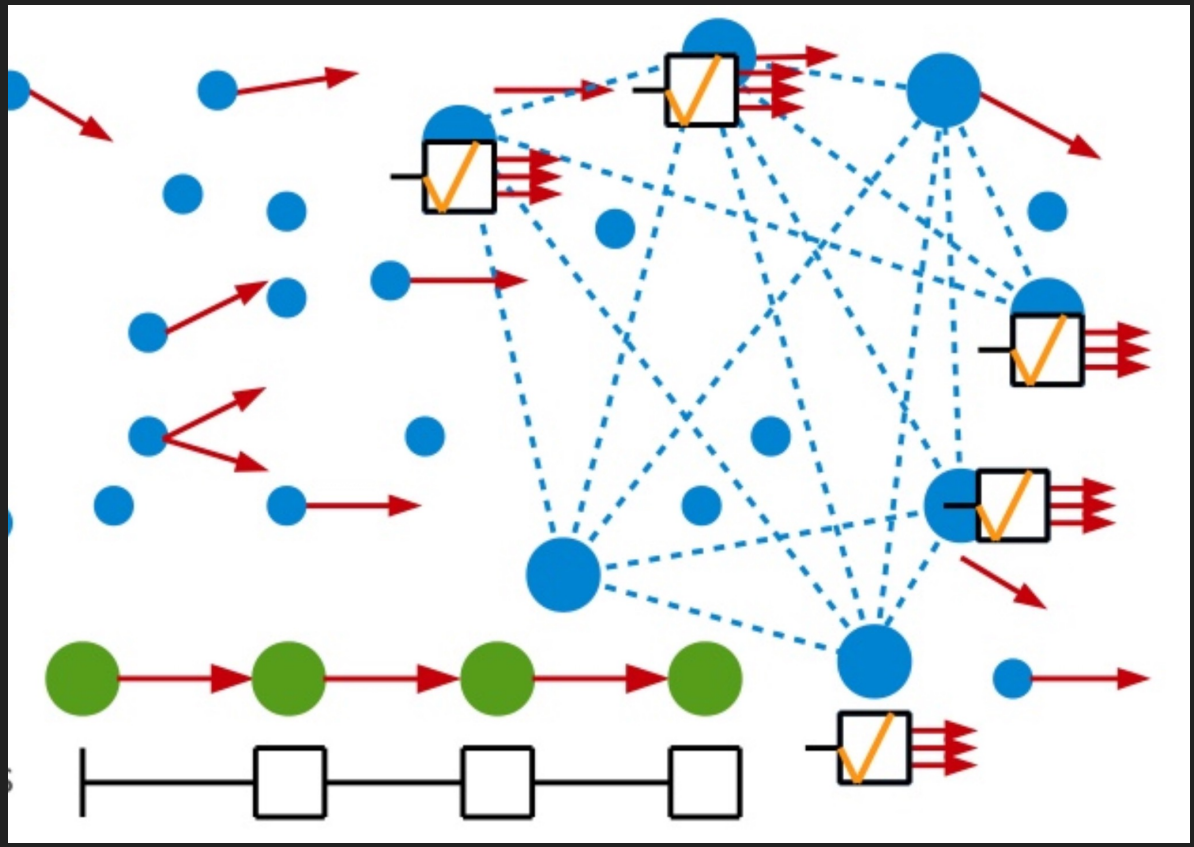


Proof of Stake

- Do not let every validator compete with each other (wasteful with resources)
- The validator that will create a new block is chosen deterministically
  - based on the cryptocurrency it puts at stake as a security deposit to guarantee the correctness of its validation
  - If validator does not show up, another validator can be selected
- No block reward, the validator of a block takes the transaction fees
- The validator will lose part of its stake if it verifies a fraudulent transaction
  - Stake must be higher than the transaction fees
- More open variants of the PoS protocol may select the validator based on
  - a combination of the transaction hash value and size of stake (NXT)
  - the number of coins and duration for which they have been held (Peercoin)
  - ...
- Advantages:
  - (Much) lower energetic consumption
  - Weight of each validator voting for a block depends on respective size of deposit, not computational power
  - Better resistance to Sybil (51%) attack (more expensive)

Consortium consensus (BFT, Hyperledger)

- Permission:
  - Designated set of validator nodes
  - Central entity controls validator group membership
- Principle: BFT/Byzantine agreement
  - Tolerates f-out-of-n faulty or adversarial nodes
  - Generalized quorums among validators
- Verification:
  - Transaction sent to validator nodes
  - Consensus protocol validates transaction, decides if valid, and disseminates result of agreement



Hyperledger Fabric

- Platform:
  - Runs smart contracts (chaincode), an arbitrary GO program within Docker containers
  - Scales better (limited node executions, independent computations in parallel)
  - Permits confidential data on blockchain
- Endorsement
  - Endorsement protects against unauthorized state
  - Every chaincode may have different and dedicated endorsers
- Transactions
  - Deploy new chaincode, invoke an operation, read state
  - All peers apply state changes in order – checks that transactions are properly endorsed
- Consensus:
  - Consortium blockchain using traditional consensus (BFT, Paxos)
  - Separate process from endorsement
  - Only requires communication (pub/sub messages) and ordering of endorsed transactions