cryptocurrencies and distributed ledger technologies

a (largely incomplete) timeline

- 1999: first popular p2p service (Napster)
- 2008: Bitcoin: A Peer-to-Peer Electronic Cash System
- 2010: first real transaction
  - 2 pizzas for 10K BTC
- 2011: “Altcoins” begin to appear
  - Namecoin, Litecoin, etc.
- 2014: UK treasury commissioned a study on cryptocurrencies
- 2017:
  - BTC quotation about 16K$
  - Russia and Estonia announce plans for government backed cryptocurrency
  - blockchain (DLT) and cyrptocurrencies regarded as **game-changers**

Bitcoin, blockchain and DLT

- Bitcoin is a cryptocurrency...
- ...based on a technology called **blockchain**
- a number of variation of the blockchain are possible and many are used
- they collectively are called **Distributed Ledger Technologies (DLT)**

a DLT solves one fundamental problem

- many subjects need to agree on **transactions**...
- ...**without trusting each other**

- transactions are recorded on a **ledger**
- the ledger is **replicated**
  - each participant has a copy of it

- **consensus** on what is a "good copy" of the ledger is reached in a **distributed** manner
  - **no central authority** to be trusted

DLT for a cryptocurrency

- transactions are payments
- the ledger records payments
- a “good copy” conforms to plain accounting rules, e.g....
  - **no double spending** of money
  - controlled **money creation**
  - **no charge back**
  - conditions to unlock funds

  ...and many other technical rules
  - e.g. format of the records

ledgers and security

- a ledger is used by a community of *subjects* (or *parties* to transactions)
- it is updated for each transaction
- requirements
  – parties to a transaction need **guarantees about recording and consensus**
  – old transactions must be **immutable**
  – all involved nodes **see and agree on a single ledger status** at a certain instant
    - ...that conforms to all consensus rules
- DLTs fulfill these requirements without centralized trusted authority

potential applications of DLT

- real estate registry
- companies registry
- parcels delivery tracking
- civil registry
- financial transactions
- insurance
- medical records
- trial records
- ...

many have legal implications

(un)permissioned DLT

- unpermissioned DLT
  - anybody (as subject or node) can join
  - large networks
  - slow
  - e.g., Bitcoin
- permissioned DLT
  - only authorized subjects/nodes can join
  - small networks
  - fast
  - typically belonging to industry/banking consortiums

cryptocurrencies elements

- **identifiers** of transaction parties (addresses)
- **ledger** content, format, consistency
  - many technical rules
- **p2p protocol** to broadcast accepted and pending transactions
- **distributed consensus algorithm**
  - a way to reach consensus “securely”
- **incentives**
- **money creation and accounting constraints**

Distributed Ledger Technology (DLT)

- **identifiers**
- **ledger**
- **p2p protocol**
- **distr. consensus alg.**

DLT permissioned

- **incentives**

DLT unpermissioned

- **money creation and accounting constraints**

cryptocurrency

identifiers

- identification of subjects is done by private/public key pairs
- in unpermissioned DLT, subjects autonomously create private/public key pairs, possibly many of them
  - having many IDs improves confidentiality
- in permissioned DLT, subjects might be all well known to all nodes
  - shared subject directory and strictly regulated access

ledger

- essentially a log of transactions
- addition of transaction occur on a **block** basis
  - a block contains many transaction
- most of the machinery of a DLT is about the addition of a block to the ledger

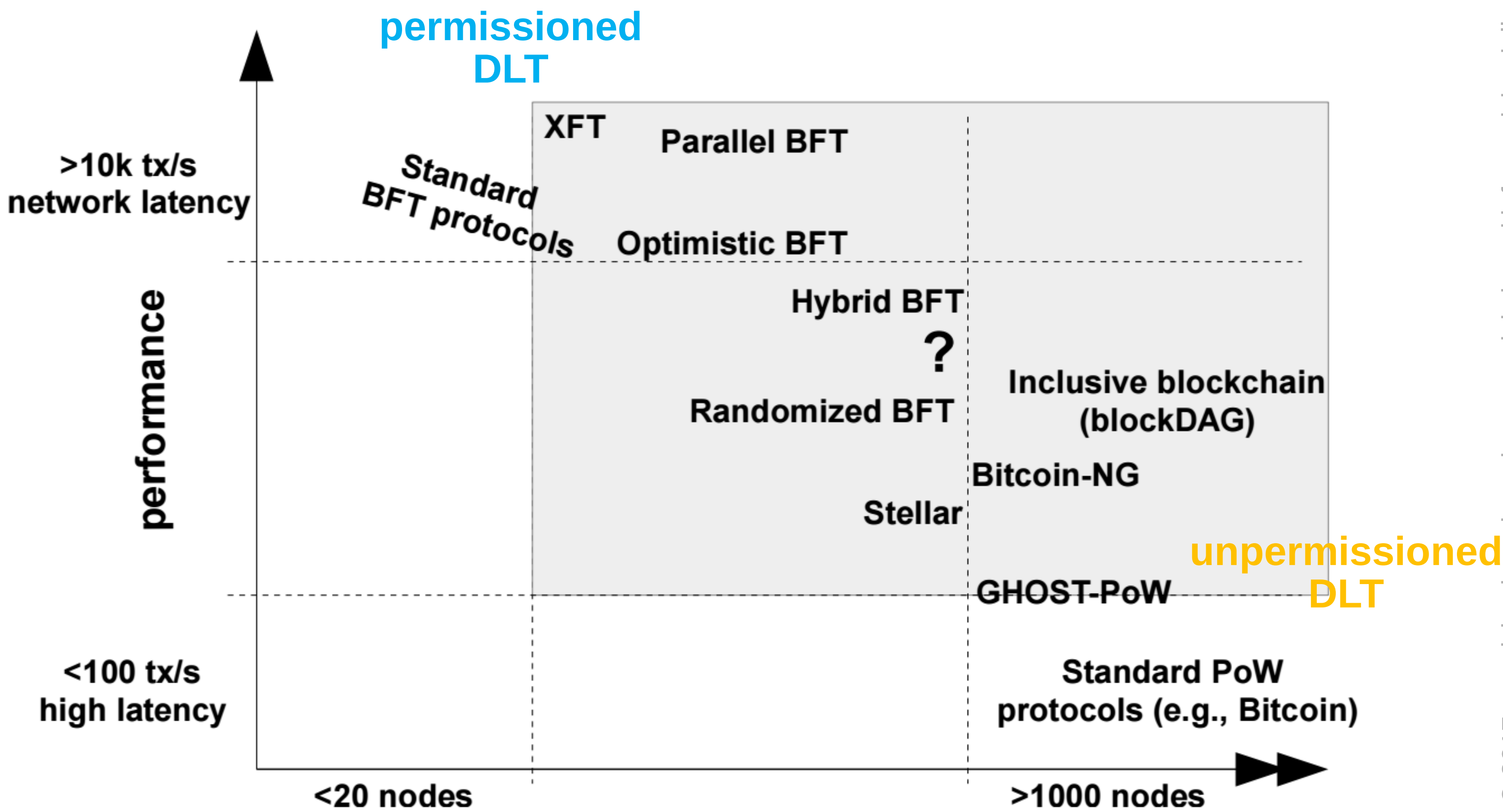
p2p protocol

- nodes discovery (the first to connect to)
- node interconnection
- broadcasting
  - new blocks added to the ledger
  - pending transactions
- each new/pending transaction is broadcasted
- when a node pretend to add a block to the ledger, the new block is broadcasted

distributed consensus algorithm

- it is a way to accept a new block
- mandate that "all" accept the same block(s)
  - eventually they will have the same view of the ledger
- check for format rules and other semantic rules
  - easy
- contrast "byzantine" (malicious) behavior of nodes...
  - ... which might pretend to subvert the rules
  - hard
- many solutions, a few very famous
  - Proof-of-Work (for unpermissioned DLT)
    - slow but scale to high number of nodes
  - Byzantine-Fault-Tolerant (for permissioned DLT)
    - fast but feasible only for a small number of nodes

distributed consensus algorithms overview



source: M. Vukolić. The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication. iNetSec 2015 (adapted)

incentive

- needed only for unpermissioned DLT
- anybody can join the DLT
  - usually is better to have a large number of nodes
- people have to get an advantage to join
  - joining means sharing resources with a community
- the advantage is usually some form of “money” (*tokens*)
  - that is, even not strictly money-related DLT have their own form of currency that can be exchanged for real money

"money semantic"

- creation
  - mining, premining, minting, etc.
  - tightly related with the incentive problem
- accounting rules
  - no double spending
  - no charge back
  - transaction fees
- unlocking of funds
  - proving ownership (by cryptographic means)
  - possibly complex rules (smart contracts)

bitcoin

relevant concepts

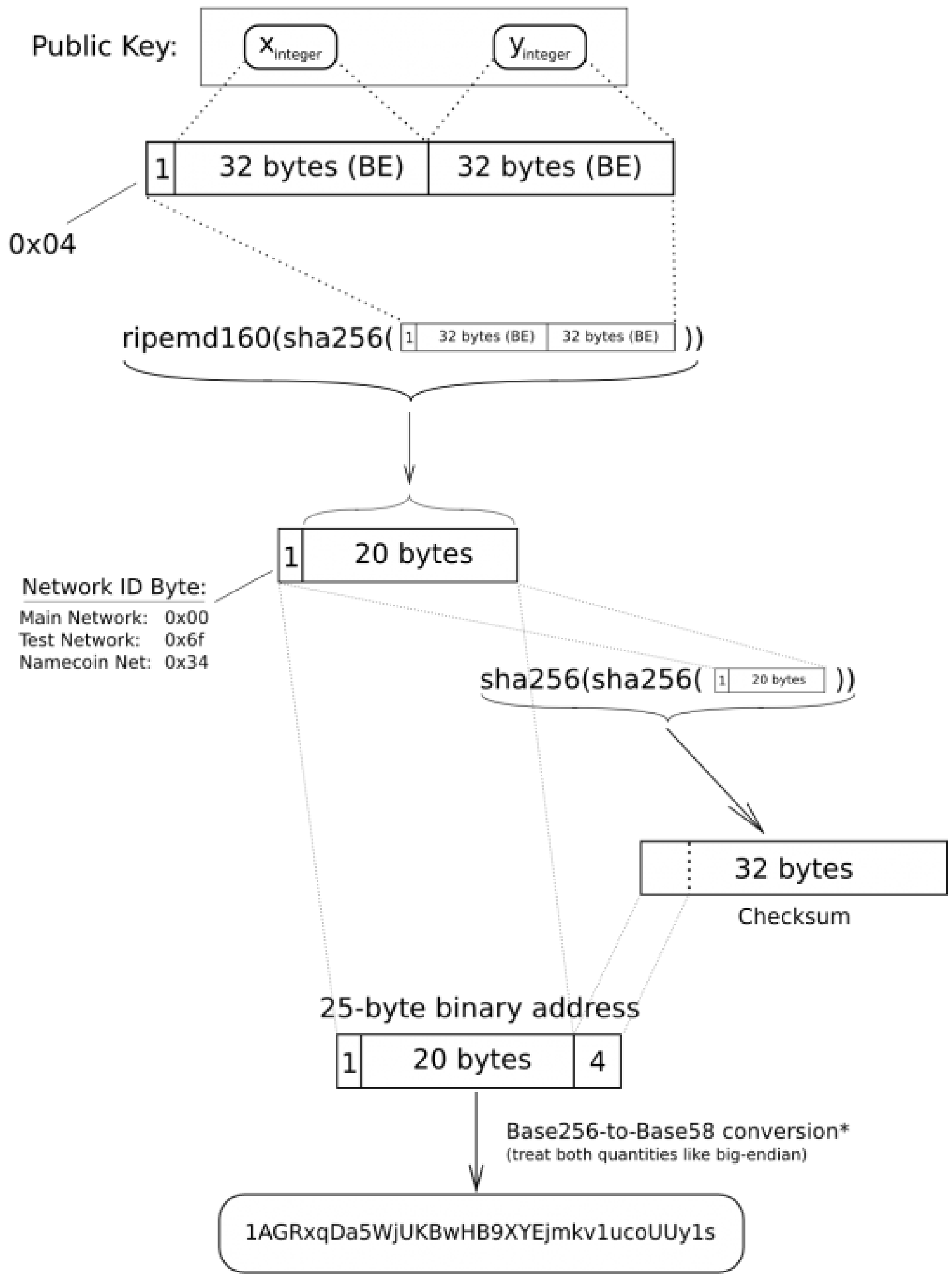
- addresses
- transactions
  - txin, txout, utxo, fees
- blocks
- blockchain
- proof-of-work

addresses

- created off-line by your wallet software
  - as many as you want
- private/public key pair
- **an address is a cryptographic hash of the public key**

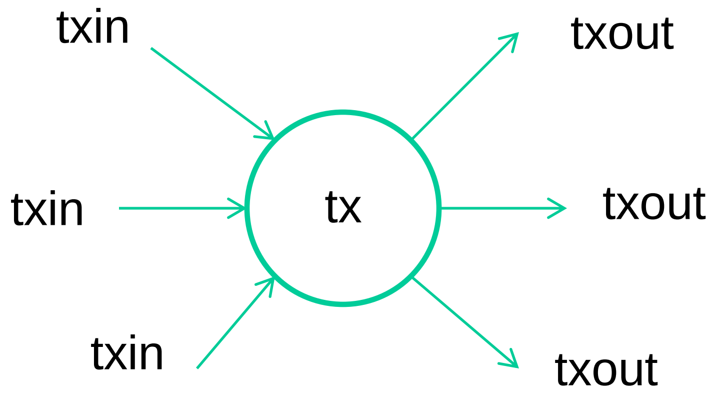
- ECDSA standard is used
- notable properties:
  - private keys are random numbers
  - the public key are derived from the private one
  - password based wallet with no explicit key storage are possible
    - Hierarchical Deterministic wallets

address derivation details



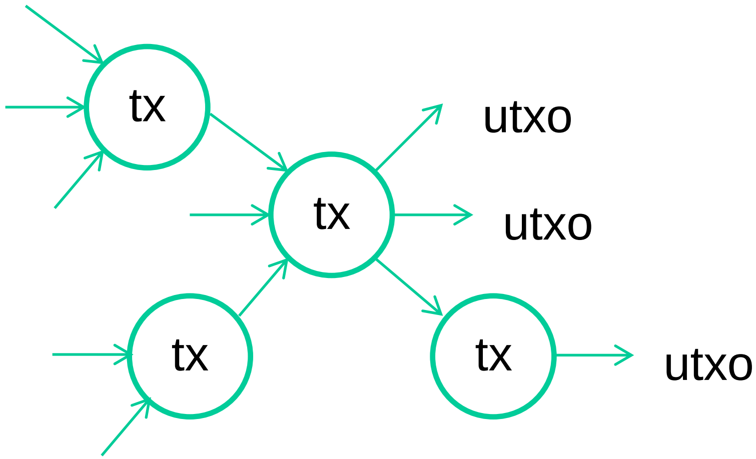
- this is the most common kind of address
- it is called “pay to public key hash” (p2pkh)

transactions (TXs)



- transactions form a directed acyclic graph
- txout is associated with...
  - an **amount**
    - espressed in satoshi
    - 1 satoshi = 10^-8 BTC, i.e., about 0.0001$
  - a **destination address**
    - actually a script typically checking for the address
    - dest. addresses may or may not belong to the same subject

utxo



- a txout can be...
  - **spent**, i.e. attached to a txin of another transaction
  - **unspent**, called **unspent tx output (utxo),** i.e., no txin attached
- currently "existing" bitcoins are those "stored" at utxo
  - ... and at addresses associated with current utxo
- **a txin always spends the whole utxo amount**
- partial spending is realized by adding a txout with a "change address"
  - i.e. returning money to addresses that belong to the same subject owning addresses involved in txin

transaction (un)balance and fees

- sum of amounts for txin's should be greater than the sum of amount of txout's
- the difference is the transaction fee
  - it is implicitly specified by the unbalance

$$Fee = \sum TxIn - \sum TxOut \geq 0$$

- the fee goes to the node that succeeds in putting the transaction in the blockchain
- nodes pick transactions with the highest fees!
  - block size is limited to 1MB! (see after)
  - your transaction might never be accepted due to low fee

txid

- a txid is a cryptographic hash of a transaction
- it is “almost” an id
- “almost”?
  - a design mistake
  - security problems was fixed
  - you can safely consider it as an ideal id

transactions: getting money out of a utxo

- txout are ordered
- each txin specifies a txout by...
  - txid (the transaction)
  - the index (i.e., the order) of the txout in that transaction
- each txin provides a **cryptographic proof** that the tx creator has the private key for the destination address of the txout

getting money out of a utxo: cryptographic proof

- this is similar to a challenge response protocol
- txin of a transaction tx provides...
  - public key whose hash should match the address in txout
  - **signature** of a string X
- X is a string derived from...
  - tx where signatures are omitted
    - signing the signature is clearly impossible!
  - the destination address contained in referred txout
    - actually a string derived from the script containing the destination address!
  - it is a quite tricky procedure
    - see https://en.bitcoin.it/wiki/OP_CHECKSIG

lifecycle of a transaction

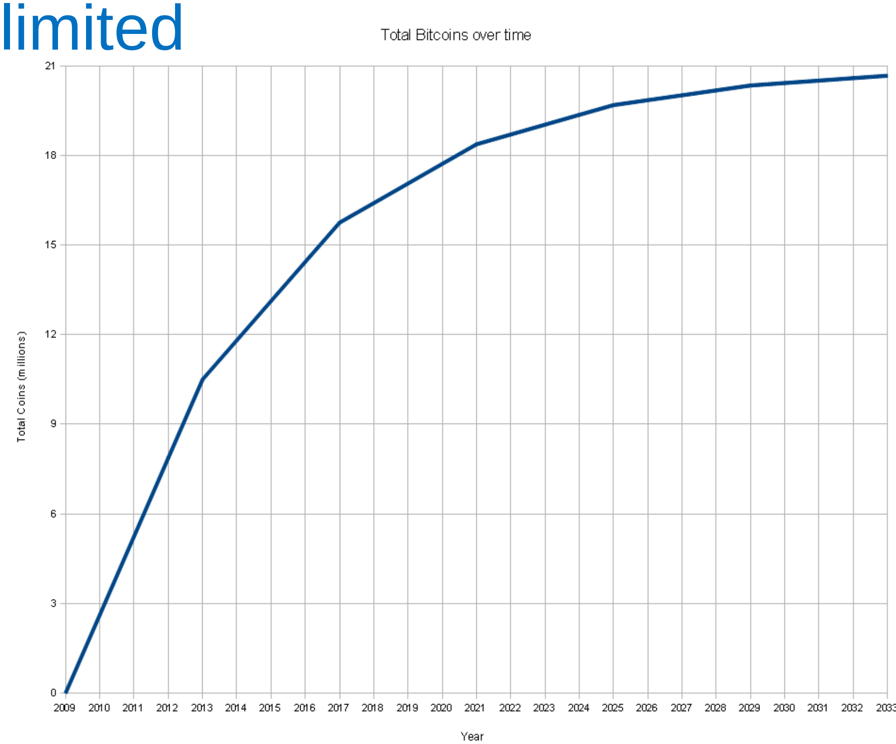
- a node n creates a tx locally
  - it computes all signatures proving private key possession
  - the node should know all previous transactions
- n send it in broadcast
- nodes that receives tx check for its validity
- all nodes puts tx into a “pool” of pending transactions
- all nodes try to put tx in the **blockchain**

blockchain

- this is the ledger of bitcoin
- it is made of **blocks**
- a block contains many of transactions
- blocks are chained in a sort of authenticated singly linked list
  - hence, blocks are strictly ordered and numbered (depth of a block)
- adding a block is...
  - difficult (proof-of-work approach)
  - provides the node with a reward (incentive) of newly created bitcoins and transaction fees

reward (Bitcoin creation)

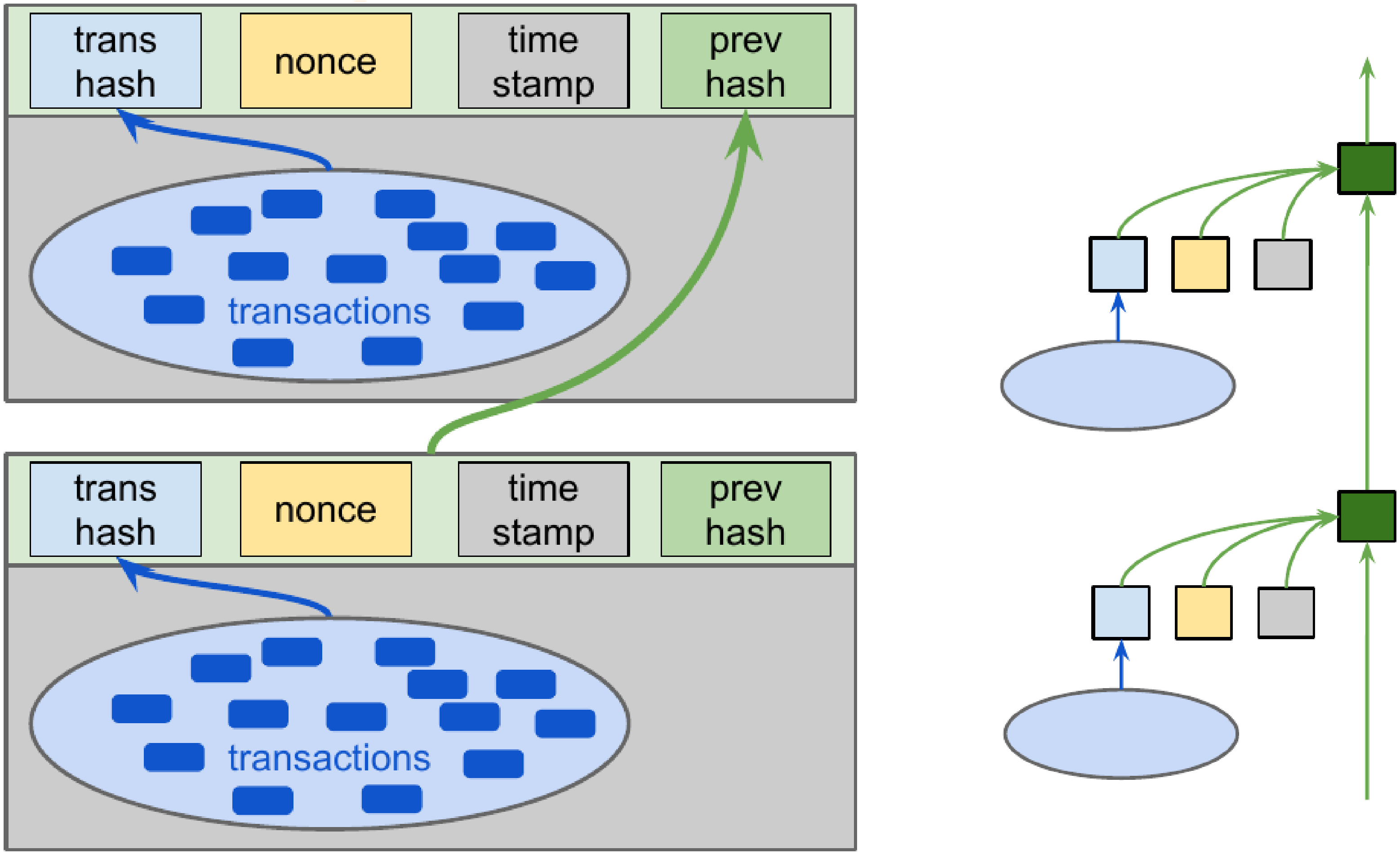
- each block create a new amount of bitcoin
  - called “coinbase”
- started at 50BTC/block
- halved every 210000 blocks (about 4 years)
  - total number of BTC is limited
- as of Dec 2017 it is 12.5BTC/block (about 200K$)
- it is represented as a special transaction
  - the first of each block, no txin, only one txout



block content

- payload, i.e., the transactions
  - block size is limited to 1MB
  - nodes pick transactions with the highest fees! your transaction might never be accepted due to low fee
- header
  - timestamp (very roughly approximated)
  - hash of all transactions
    - **a root hash of a Merkle hash tree** of all transactions in the payload
  - the hash of the header of the previous block of the blockchain
  - a nonce
    - this is the solution of the puzzle for the proof-of-work approach
  - other stuff

block content



courtesy of G. Di Battista and R. Tamassia

consensus

- adding a block requires to solve a cryptographic puzzle (proof-of-work, PoW)
  - by enumeration approach
- **consensus is implicit**
  - a node that works for the next block is accepting all previous ones
- **forks** may happen:
  - two nodes solve the next block at roughly “same time”
    - with two distinct solutions
  - the two block are broadcasted (fork)
    - actually some nodes see only one of them (non instantaneous broadcast), others see both and choose one
  - the two chains might grow independently for a while

fork resolution the longest chain rule

- a node that sees more chains **chooses the longest one**
  - transactions that are in a discarded block are put in the pending transaction pool again
- the longest chain has more work done on it
  - in terms of computation performed
- the chain that grow faster is random

consensus attacks general objectives

- changes to old blocks already accepted by at least some nodes
  - it is about integrity of the blockchain: important for all DLTs
  - might allow chargeback, double spending, and illegitimate change other parameters of the network
- DoS: denial of acceptance of certain transactions

transaction confirmation

- **confirmed: stored in an immutable block, forever**
- PoW does not provide “mathematical guarantee” of confirmation

- a transaction is considered confirmed if it is enough deep in the blockchain!
- “enough” depends on the criticality of the transaction
- usual confirmation depths are 1 to 6

consensus attacks and confirmation depth

- changing of a deep block *b...*
- ...requires the attacker to solve again all blocks above *b*
- the attacker needs a huge amount of computing power to reach and surpass the legitimate chain

- the more *b* is deep the more is "confirmed"

consensus attacks: Sybil

from "Sybil: The True Story of a Woman Possessed by 16 Separate Personalities" –F. R. Schreiber - 1973

- who controls a large number of nodes can isolate a "victim" node
- the victim see a different blockchain where she can get "malicious payments"
- the malicious payment disappear when the attack terminates and legitimate chain is broadcasted
  - chargeback, double spending
- can be detected by observing an anomalously low "hash power"

consensus attack: 51%

- who controls more than 50% of the computational power can...
  - disconfirm recently confirmed blocks
    - by surpassing with its chain all other forks
  - get 100% of the rewords
    - by keeping adding blocks

proof-of-work: the puzzle

- find a block whose header hash is below a certain **target threshold**
  - SHA256(SHA256(Block_Header))<threshold
  - lower is harder
  - difficulty = maxthreshold/threshold
- target threshold is “given”
- a node can search for a solution varying...
  - nonce
  - timestamp (within certain limits)
  - the set of transactions

target threshold adjustment

- the target threshold at a certain instant is fixed for all nodes
  - current target is stored in the last block
- it is adjusted so that time for solving the puzzle is 10 minutes on average
  - the average tx acceptance delay tend to be 5 minutes
- it is a feedback control loop
  - inputs: the time needed for last 2016 blocks and current threshold
  - output: new threshold
- adjustment happen every 2016 blocks
  - two weeks on average
  - only the node that solve the k*2016th block can change it (it is a consensus rule)

maximum theoretical transaction acceptance **throughput**

- maximum size of the block is 1MB
- minimum useful tx size is 226 bytes
  - tx with p2pkh addresses, two outputs (one for change), one input
- 1 [MB/block] / 226 [B/tx] / 600 [s/block] =

7.32 [tx/s]

- current average about 3.5 [tx/s]

bitcoin policy: the block length dilemma

- larger block size
  - lower fees
    - more txs in a block
  - harder to be a miner
    - more bandwidth, more ram, etc.
  - less democracy
    - limited number of miners can easily decide on the future of Bitcoin: easier to agree to change rules, easier to collude to reach 51% computing power
- smaller block size
  - higher fees
  - easier to be a miner
  - more democratic governance

segregated witness (SegWit)

- a soft fork activated on August 24, 2017
- strip signatures from transactions
  - put it in a “side chain”
  - replace the concept of block length with “block weight”
- new address format (SegWit address)
  - transaction should get money from new segwit addresses to “weight less”
  - slow adoption
- equivalent to have about 2MB of block size

Simplified Payment Verification (thin clients)

- thin clients do not store the whole blockchain
- they store just block headers
  - 80 bytes, about 4MB/year
- when transaction information is needed an untrusted full node is contacted
  - Merkle tree! proof used for integrity check against the root hash stored in the trusted header