

Decentralized Systems Engineering

CS-438 – Fall 2023

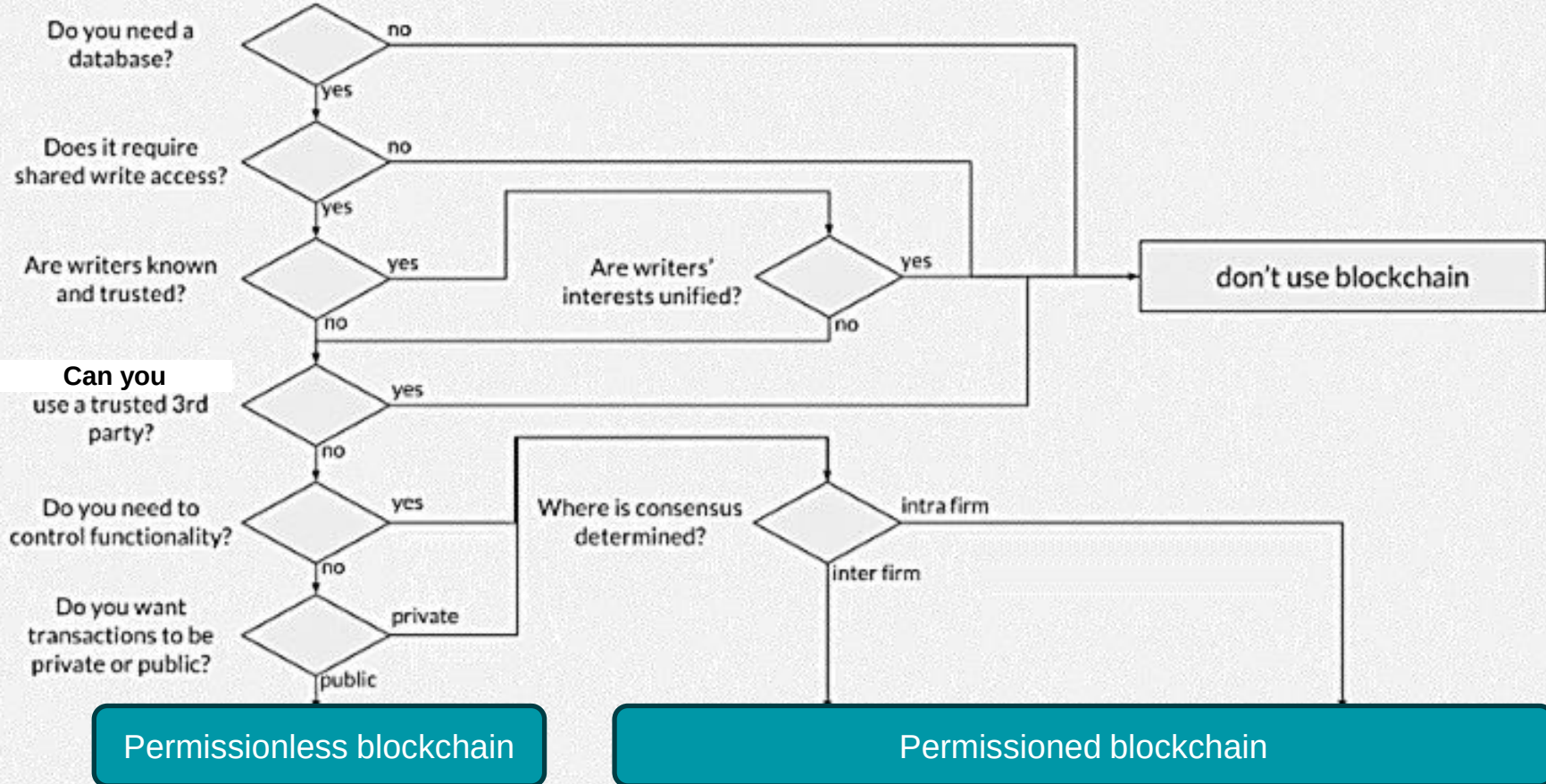
DEDIS

Pierluca Borsò-Tan

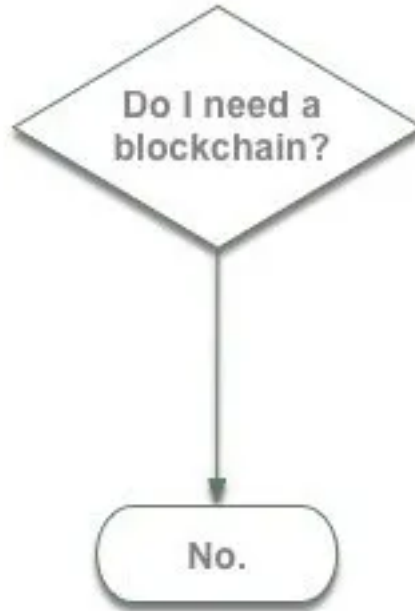
EPFL

Credits: B. Ford

Do you need a blockchain ?



Do you need a blockchain ?



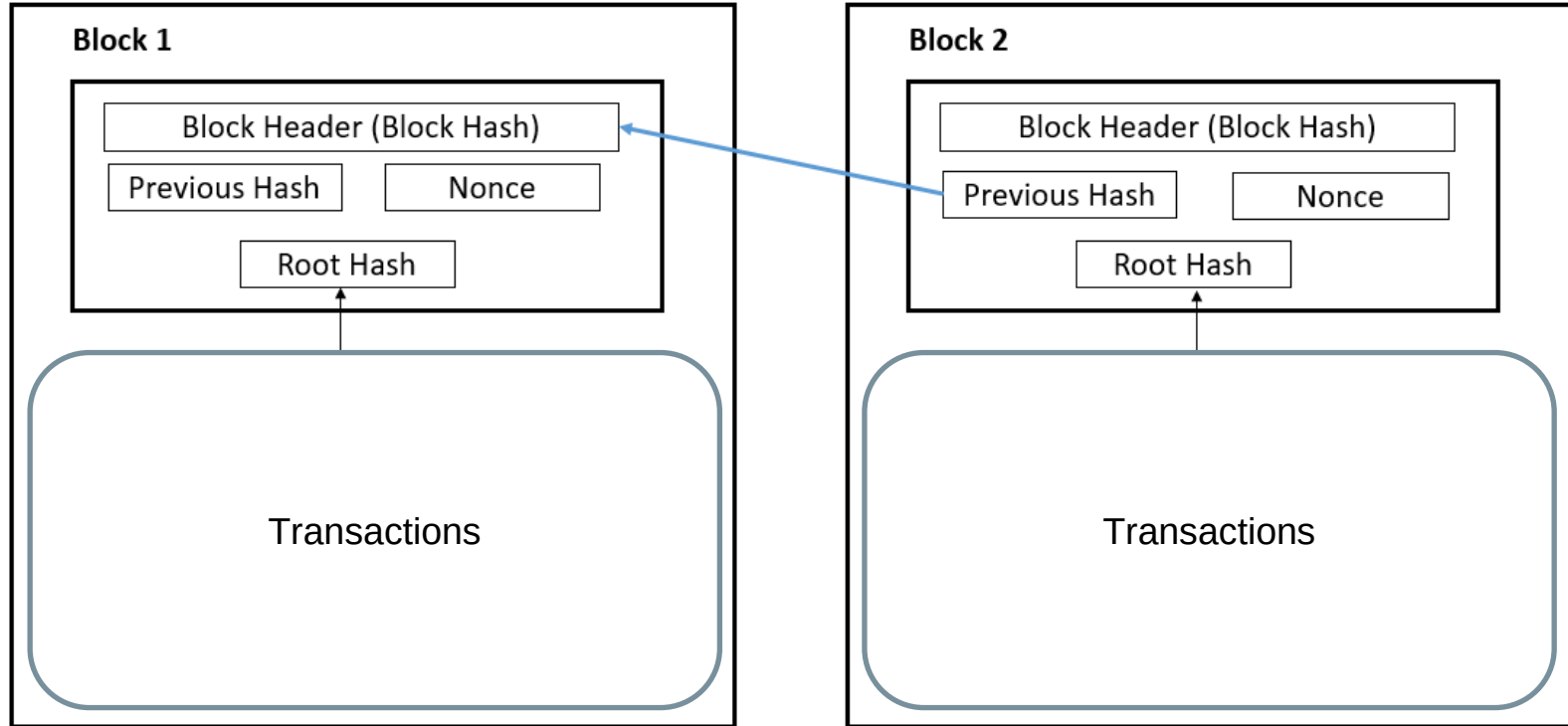
Bitcoin

Revisited

Goals

- Currency with no trusted, central authority → Permissionless Consensus
- Works online (like a credit card)
- Anonymous / pseudonymous (“like cash”) → Cryptography
- Non-reversible transactions
(probabilistic finality is OK) → Append-only (“block chain”)
- Conditional payments (like contracts) → Later today ;)

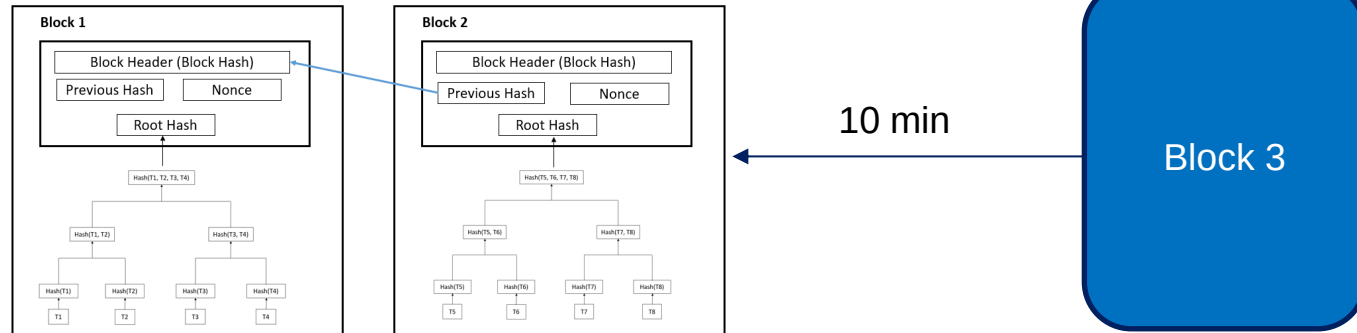
Blockchain structure



Key ideas

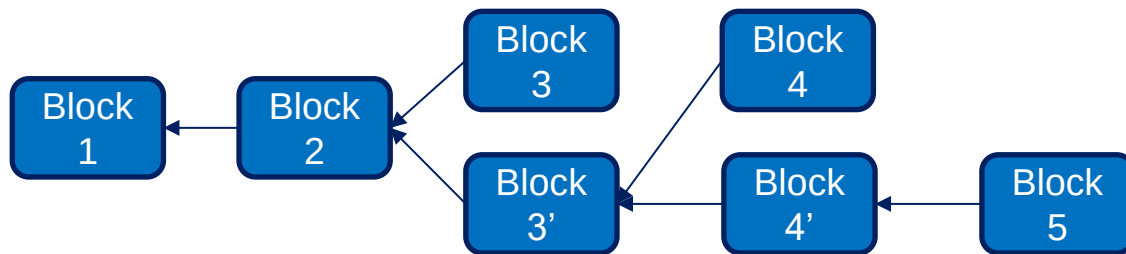
Proof-of-Work

- “Miners” solve computational puzzles (hash with leading N zeros)
Computational power = Hash rate (H/s)
- Puzzle difficulty is adjusted to keep block rate (roughly) constant
→ compensates for changes in mining power



Assumptions

- Threshold assumption: majority of mining power is honest
... independently of the number of nodes
- Longest / heaviest chain rule
... transient safety violations (e.g. forks, reversed transactions) are OK
... eventually forks will be resolved (based on expended work) !



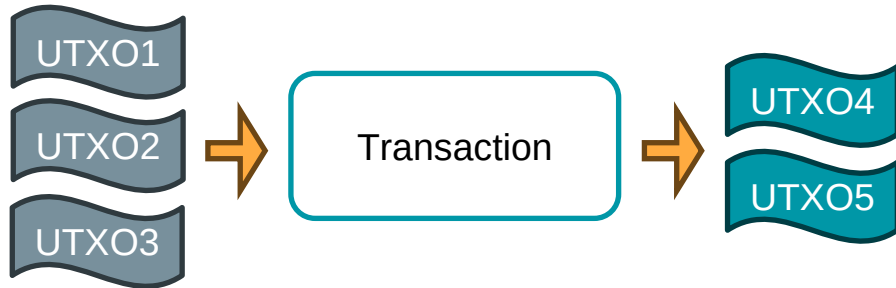
- Probabilistic finality – 6 blocks (1h)
- Economic incentive compatibility
- Network connectivity / propagation – synchrony assumption (10 min)

Transactions – UTXO vs Account

How can we model transactions ?

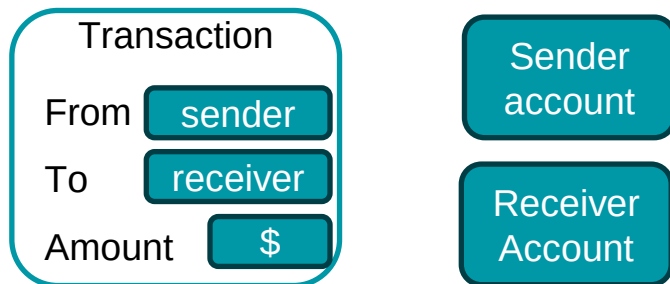
- Cash (UTXO)

→ **Bitcoin**



- Bank (Account)

→ **Ethereum (later today)**



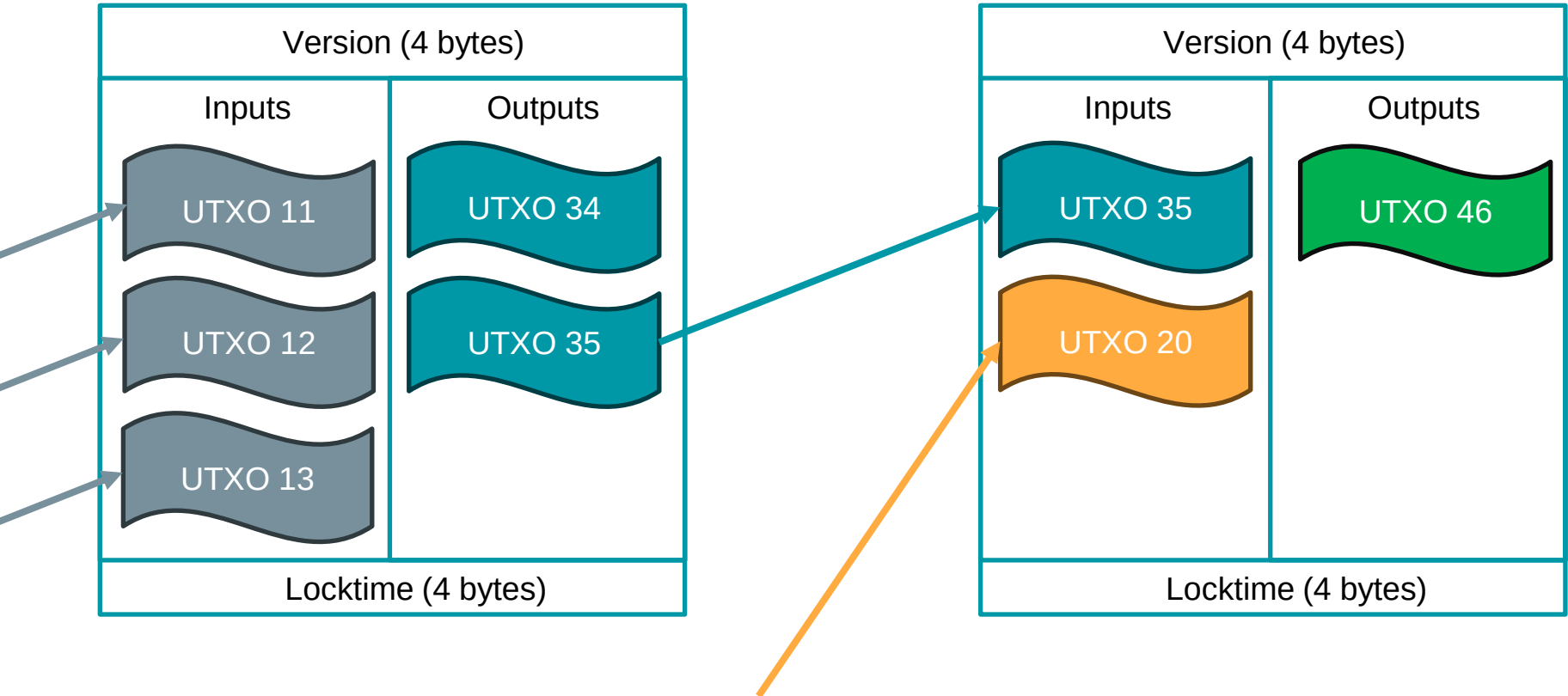
What are the advantages and disadvantages of each ?

Where are the transactions before a block ?

Bitcoin's (and other blockchains') nodes deal with two storage pools:

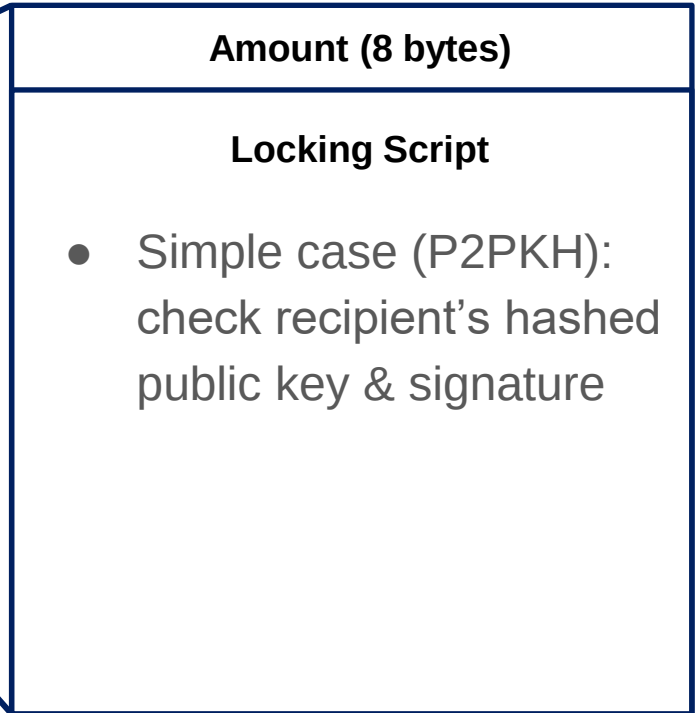
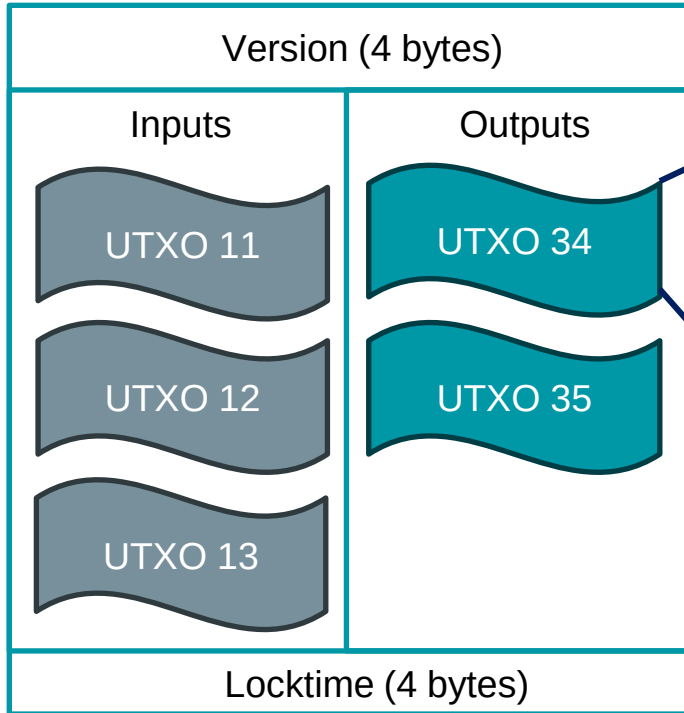
- The blockchain
 - final (probabilistically)
 - eventually, the single source of truth
- The memory pool (aka “the mempool”)
 - “in-flight” transactions
 - propagated among nodes
 - the source of transactions in a block

Bitcoin Transactions



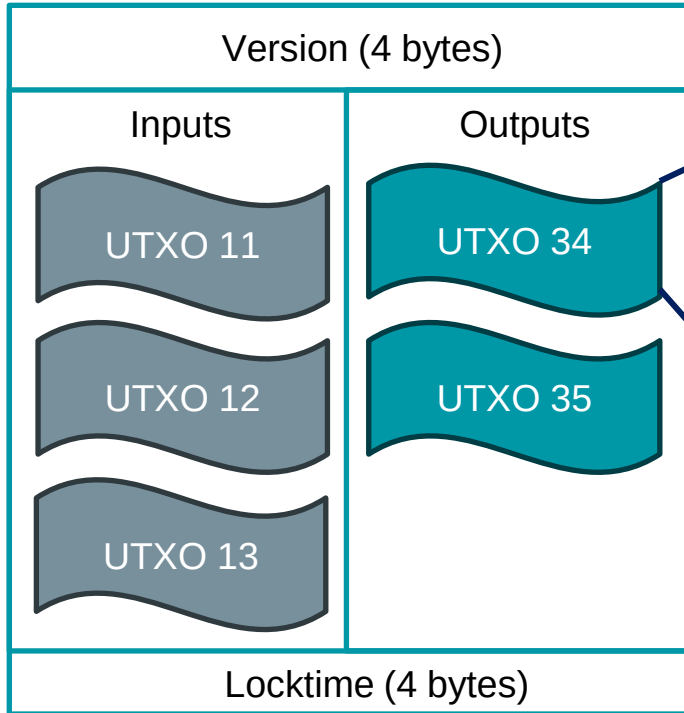
Bitcoin Transactions

“Pay to PubKey Hash”



Bitcoin Transactions

“Pay to Script Hash”

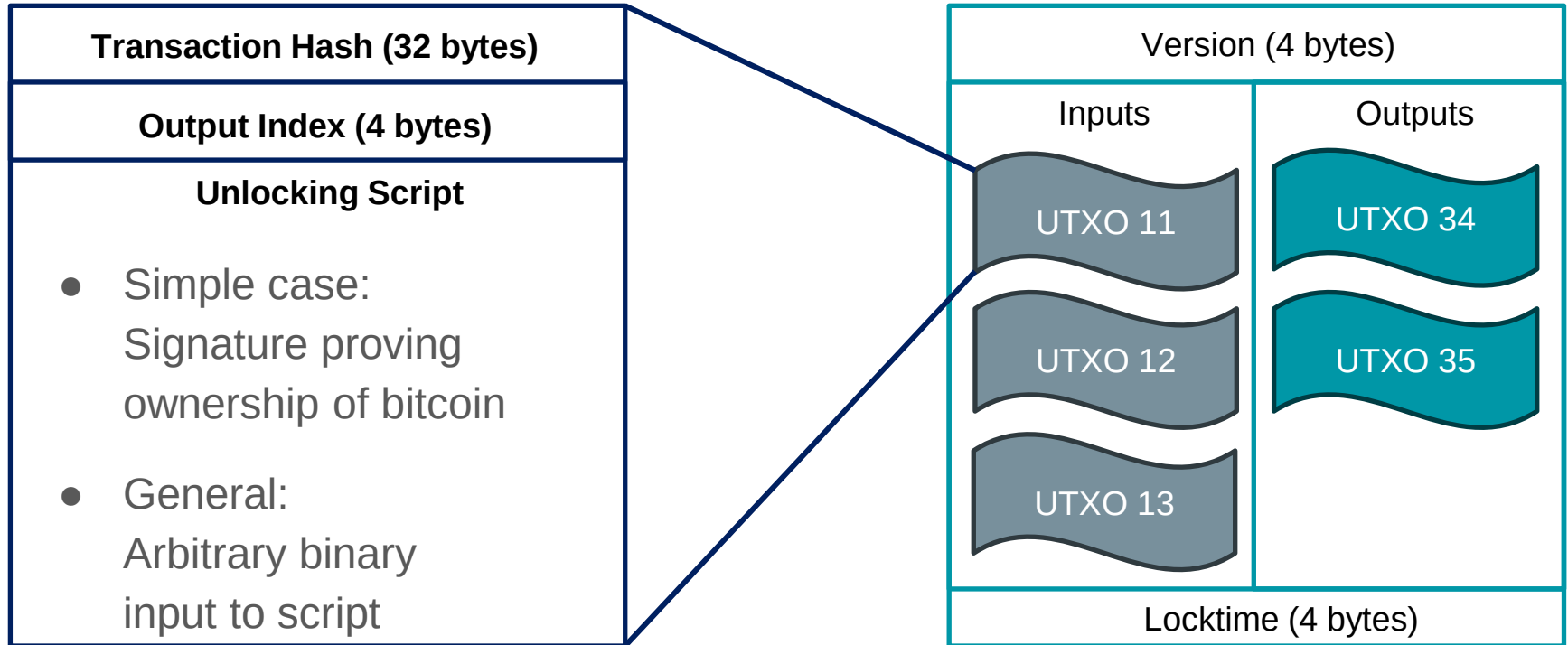


Amount (8 bytes)

Locking Script

- Simple case (P2PKH): recipient's public key
- General case (P2SH): Code to check validity of a spend request (binary output)

Bitcoin Transactions – Inputs



Bitcoin Scripting

Putting it all together ...

- UTXO destination can be a script (Lock Script)
- Script checks spending authorization
- Transaction input must provide data satisfying the check (Unlock Script)

Enables non-trivial logic:

- Multi-signatures (t-of-n)
- Time lock vaults / contracts
- Payment channels (Lightning net)
- Notaries, side-chains

Bitcoin Scripting: example

Multi-signer authorization (multi-sig)

- Any t of n co-signers can authorize spending
- Ex. logic for $t=2, n=3$

Script: $a \leftarrow 0$

$a \leftarrow a + \text{check}(K_i, T, I[0\dots 63])$

$a \leftarrow a + \text{check}(K_i, T, I[64\dots 127])$

$a \leftarrow a + \text{check}(K_i, T, I[128\dots 191])$

return $(a \geq 2)$

Scripts – Who runs them ?

- User who wants to spend
- All miners when including the transaction in a block
 - first to create the block
 - then to validate the block
- Miners need to achieve consensus in block validity
- Determinism is paramount !

Bitcoin Scripting limitations

- Only a limited number of bytecodes
- No backward branches
- Bytecode limited (1 block < 1MB, pre-SegWit)
- Completely deterministic
- Inefficiency of deterministic VM

Ethereum

Smart Contracts

Ethereum – Generalizing smart contracts

- Account-based
 - ... account persist across transactions
- Richer bytecode language
 - ... still limited
 - ... but Turing complete, with loops !

How can we deal with infinite / unbounded execution ?

Ethereum – Gas

- Deterministic, virtual execution time
... (weighed) instruction count
- Each script execution has a gas limit
... that must be paid up-front (invoker or script)
... charged based on usage
- If script succeeds within gas limit, effects yield atomic state change
- If script exhausts gas limit, no state change – but gas still charged

Smart Contracts – Applications

- Trustless Insurance – AXA Fizzy (flight delays insurance)
- New payment/finance methods
- Decentralized naming – † Namecoin (DNS-like), Filecoin, ...
- Tokenization – ICOs (Initial Coin Offerings), NFTs, ...
- Storage – on-chain, off-chain management
- Programmable markets – auctions, prediction markets, quadratic voting, ...
- Games – gambling, CryptoKitties, etc.
- Decentralized governance – DAOs
- Automated market makers – Uniswap (trade between coins)

Smart contracts – Issues & limitations

- Inefficiency of deterministic VM → eWASM
- Oracle problem
 - Trusted authority
 - Decentralized Oracles
- Front-running attacks / “Dark Forest” → Active research area
- Secrets
 - Keep secrets off-chain + zk-proofs
 - On-chain secrets (Calypso)
- Smart contract bugs (ex. “The DAO”) → Recourse ? Recovery ?
- Improvements/evolution is difficult → Permissionless innovation? Versioning?

Next steps

Optional readings:

- Ethereum: A Secure Decentralised General Transaction Ledger
- Ethereum is a Dark Forest
- The Law and Legality of Smart Contracts

→ Use Friday's session to ask questions