Blockchain Zusammenfassung

Roman Weber

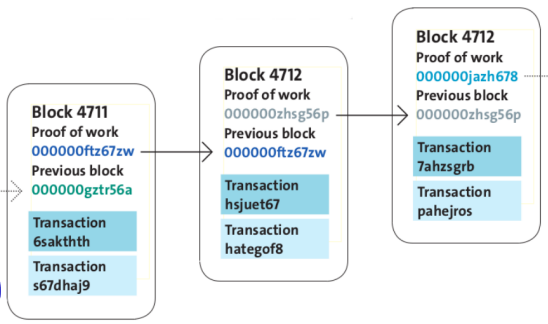
01) Repetition

Controlled Distributed Systems:
- 1 responsible organization with control
- High availability

Fully decentralized System:
- N responsible orgs / people
- Hostile environment (everybody can see everything)
- unpredictable availability

02) Repetition 2

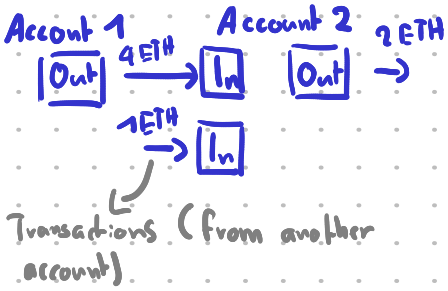
- Bitcoin is fully peer-to-peer. Validation per proof-of-work (calculate a partial hash collision → difficult to fake)
- Satoshi Nakamoto is the OG. Inventor and first miner. remains anonymous.
- A wallet has private and public key (ECDSA 256 bit)
  - private → sign transaction
  - public → receive
- Bitcoin scripting language is non-turing complete (no loops)
- Blocks have a link to previous block and contain transaction. Bitcoin has a new Block ~ 10 min (depends on chain)



- 3-6 block confirmations are considered secure
- If someone has 51% of computing power, they can exclude and modify ordering of transactions → not good lol (or double spend)
  - ↳ This sometimes happens for smaller chains. (No guarantee Ethereum or Bitcoin is not hit)
- Ethereum has loops (turing complete), New block every 14 seconds → reward always 2 ETH
  - ↳ Smart contracts

Account-based (Ethereum)

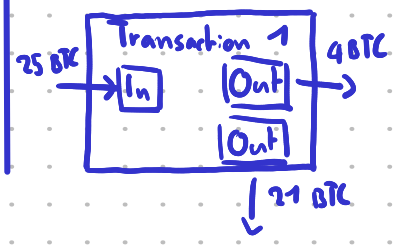
- Global state has a list with balance on each account
- Transaction is valid if sender has enough



UTXO-based (Bitcoin)

unspent transaction output

- Every referenced input must be valid and not yet spent
- All outputs are always spent



- into another Transaction or into an account

03) Solidity

Running a transaction costs GAS. EthereumVM can run opcodes, every instruction needs to be payed for. If you run out of GAS, state is reverted.

State variables to store something persistantly -> expensive to write

wei, gwei oder Ether

One contract can deploy another contract

This slide deck does not have a lot to offer, not sure what will be relevant in exam.

04) Fungible Token

Eg: Bitcoin, Ethers, CCR Token, ... ECR-20 Tokens

Three types of FT:

- Payment: Ethers, ...
- Utility: Access to certain applications
- Security: Share of real-world asset

DEX = Decentralized Exchange = Chain

Stablecoin (eg USDC): tied to real world asset

Altcoin/Shitcoin -> non-mainstream

Native -> Ethers
non-native -> self-created or USDT (contract driven)

ERC-20 standard found on OpenZeppelin

- 6 conditions (by Jan Lansky)
  1) The system does not require a central authority, distributed achieve consensus on its state.
  2) The system keeps an overview of cryptocurrency units and their ownership.
  3) The system defines whether new cryptocurrency units can be created. If new cryptocurrency units can be created, the system defines the circumstances of their origin and how to determine the ownership of these new units.
  4) Ownership of cryptocurrency units can be proved exclusively cryptographically.
  5) The system allows transactions to be performed in which ownership of the cryptographic units is changed. A transaction statement can only be issued by an entity proving the current ownership of these units.
  6) If two different instructions for changing the ownership of the same cryptographic units are simultaneously entered, the system performs at most one of them.

04) Stablecoin (eg. USDC, USDT, ...)

Stablecoins are used to counter the volatility of cryptocurrencies.

Types of backing:
- FIAT currencies (USD, EUR)
- Asset backed (gold, ...)
- crypto-collateralized (backed through other cryptocurrencies)
  - ↳ Decentralization ↗
  - ↳ algorithmic stablecoin -> Challenge Task

Epic Fail Terra/UST: goal was to have the same Kurs as USD, this was achieved algorithmically. Problem was there was so much Luna minted that it became practically worthless and UST couldn't keep its value.

↳ Total Eclipse ~~of my heart~~ of the coin

Stablecoin asset-backed CeFi (central Finance Institution):

- 50 USDT are 50 USD
- 20 USDT are bought for 20 USD → 20 USDT minted (70 USDT in total)
- 30 USDT are sold → 30 USDT burned

→ 1 to 1 ratio, bank run that might not have enough liquidity

Stablecoin crypto-backed DeFi (decentralized Finance Institution)

- when other cryptocurrencies are used. I believe this is what we implemented with the pool on uniswap.

Why use a stablecoin?

- holder does not need to pay interest
- creator of the stablecoin gets an interest rate from the low-risk investments he can do from the liquidity of his stablecoin
- Smart contracts can be implemented to sell a currency after it is below a certain limit

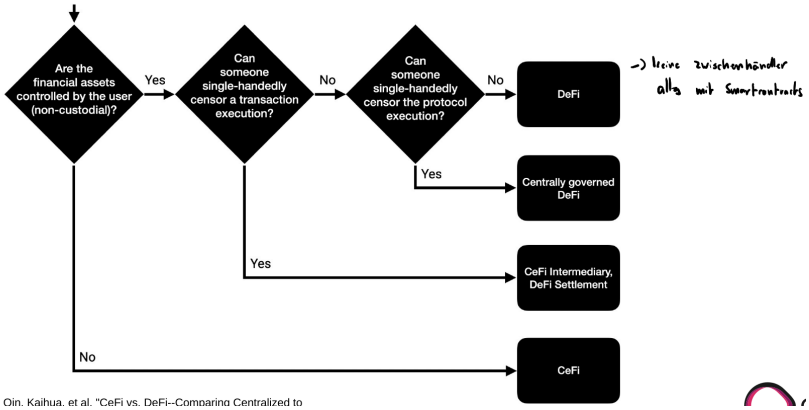
5) DeFi (Decentralized Finance)

TradFi = Traditional Finances like bank 'n shit

Key Features DeFi:

- Transparency (everything is public, no deals behind closed doors)
- Control: given to users over all assets
- Accessibility: only a computer and know-how needed to join (I find this argument questionable)

- The boundaries of DeFi and CeFi not clear cut



Qin, Kaihua, et al. "CeFi vs. DeFi--Comparing Centralized to

Key DeFi (Decentralized Financial Institution)

- Public Verification: While the app (source code) might not be open-source, all transactions are
- Custody (Sorgerecht): user controls his assets, but also bears technical risks (and scams)
- Privacy: non-privacy via smart contract on Blockchain. Or pseudo-anonymity as you can see all transactions of a user but not necessarily who he is.

- Atomicity: actions are either all executed or all aborted. Flash loan example. In a CeFi environment this can be achieved with contracts.
- Execution Order Malleability: No centralized entity ordering transactions
- Transaction Costs: Prevention of SPAM.
- Anonymous development and deployment.
- Non-shop market hours

Regulations: KYC = Know Your Costumer, AML = Anti money Laundering

- Uncertain if a developer is responsible for KYC/AML. Arguments for both.
  a) Miners can decide to temporarily transactions
  b) Blacklist to block adresses are sometimes implemented → USDT / USDC have this

CEX: trade on stockmarket. See a price, submit but until executed price might have changed

DEX (decentralized exchange): Exchange pair of coins - Uniswap pool
- larger swap have a big price impact.
- Transaction example in slides but skipped for the moment. Swaps were implemented in challenge task.

Sandwich attack: Transaction before and after swap to profit from price change
↳ This has happened

Arbitrage bots: swapping in multiple pools when there is profit to be made.
↳ Keep the same price across exchanges
↳ DEX does not work without these bots

Flash loans: Get loan and pay it back in the same transaction but you get to keep the profits.

If there is a different price across exchanges a flash loan can be used. The coins are not actually needed, if it works you keep the profit. If not, transaction is reverted.

Liquidity providers: someone needs to fill the pool with Liquidity

- LP provides (or removes) Liquidity but price does not change
- LP gets paid in liquidity of the pool that are made from Transaction fees and alike.
- Price shift loss can happen. = (same thing)

look at this again

Impermanent loss occurs when the price of tokens in a liquidity pool changes compared to when they were deposited, causing the liquidity provider to have less value than if they simply held the tokens. For example, if ETH's price increases, arbitrage traders adjust the pool's balance, leaving the provider with more DAI (stable value) and less ETH (increased value), resulting in lower total profit than holding ETH directly.

6) DAO - Decentralized Autonomous Organization

Community led entity with no central authority

- autonomous and transparent
- Smart contracts define rules and execute rules

Simplified workflow:
1) Members create a proposal
2) Members vote on the proposal
3) Proposal gets executed (if accepted)

History of DAO (Decentralized autonomous org.)

- First one was a disaster. 150 Mio $ raised but attackers drained 3.6 Mio of ETH due to a flaw in the smart contract.

Function withdraw was called again before the amount had been deducted.

=> DAO hack adress blacklisted with a hard fork in 2016.

DAO vs. Traditional Organizations

- Advantages
  - Decisions by individuals rather than central authority
  - Encourages participation
  - Public: everything is transparent and visible
  - Minimum requirement to join, is an Internet connection
- Disadvantages
  - Decisions and voting takes time
  - Currently only tech-savy people participate
  - Bridging blockchain with real world
  - Security considerations

	Corporation	Cooperative	DAO
Management	Board of directors	Board of members	Token holders
Ownership	Shareholders	Members	Token holders
Supervision	Supervisory board	Supervisory board team	Curators
Workforce	Employees	Members	Contractors

6) NFT (Non-Fungible Token)

ersetzbar

Implemented with a smart contract, a program executed in the blockchain context.

ERC standard eg the BoredApeclub

→ done via Remix IDE and Metamask.

7) POAP (Proof of attendance)

Decentralized way to prove and verify participation or attendance of events

Requirement:
1) minted through the POAP contract
2) must contain metadata (date, title, desc.)
3) come with an image

Attendee can scan QR code → POAP token in their wallet
↳ cryptographic proof

7) Wallets (and Seeds)

Wallet has a private key: sign transactions
public key: get tokens, interaction

Different type of wallets:
- Hardware
- Software (Metamask, ...)
- Paper wallet (physical document)

also there are Hot and cold wallets

BIP39 is the standard for seed phrases, 128 or 256 bit (↗ a bit short)
word list with 2048 entries of 11 Bit. 12 words · 11 bits + 4 bit checksum
random number generated, first 11 bits are looked up in word list
24 words (·11 bit) + 8 bit exist aswell

BIP44 specifies which chain is used

Good practice to write seed phrase down (even laminate lol)

8) DAI Stablecoin (algorithmic stablecoin pegged to $)

Maker DAO → mehrere Milliarden $ in real estate, staatsanleihen, ...
Verschmelzung DeFi and "normale" Finanzen
DAI Fees: Stability, DAI saving rate, Liquidation penalty
Governance via Maker DAO
DAI balancing automatisch, Emergency Measures to shutdown if needed

8) HLTC Cross-chain atomic swaps

↳ Von Ethereum zu Blockchain
↳ Either happens or does not

You want to swap Ethereum for Bitcoin
obvious way to do it is via a centralized exchange (like Binance)
"Problem": You need to trust the exchange → Exchanges can crash and have done so in the past.
↳ Also they have been caught to trade against costumers to boost their numbers.

Thus atomic swaps are used. Coins are never in an exchange, you always have full control.
↳ no trust in centralized platform needed

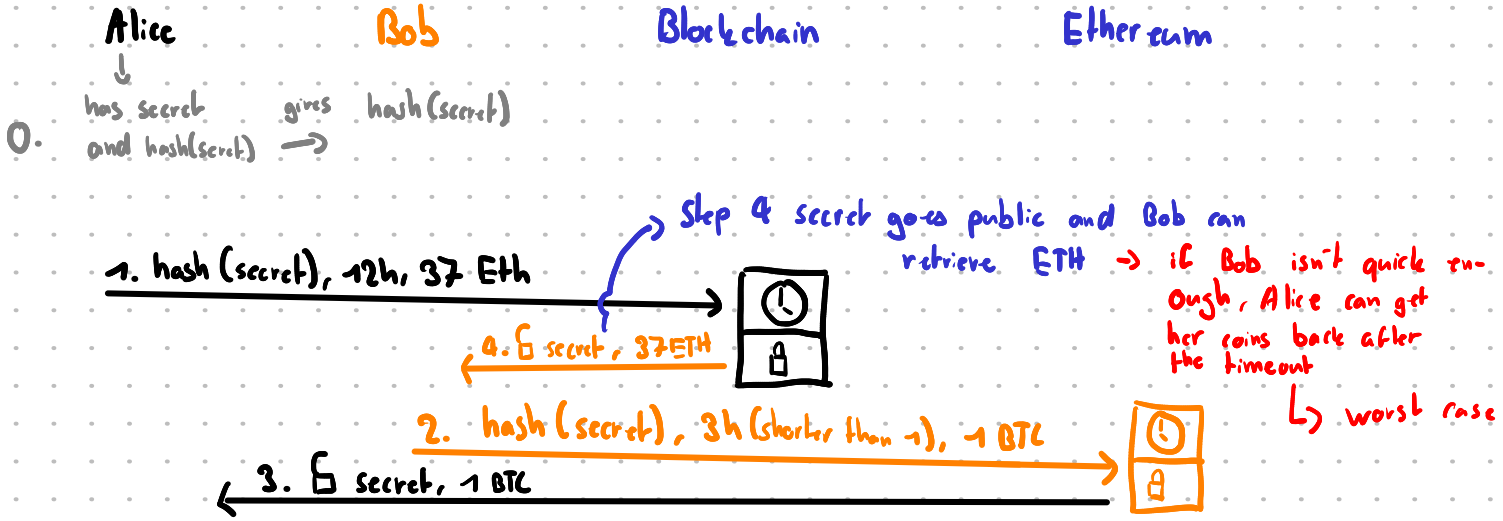
Hash Time-locked contract (HLTC)
↳ building block of cross-chain atomic swaps

They use cryptographic hashing

Hash time-lock:

- store hash secret publically in smart-contract → only unlocks if secret is provided publically.

OR unlocks automatically after timeout.



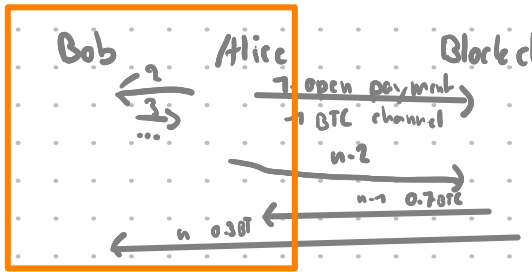
- Coins can't be stolen as two address is fix in smart contract.
- Atomic: If time runs out → nothing happens

8) Payment Channels

Blockchains grow linearly → each transaction is stored, gets more expensive for miners, compared to VISA, the chains handle very few transactions

Therefore it is vital to keep the chain as small as somehow possible.

Direct Payment Channel (with 2-of-2 multising contracts)

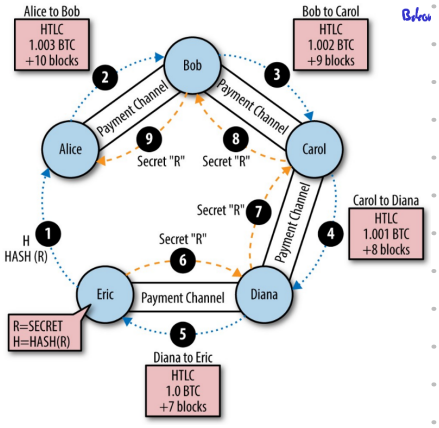


orange box is off chain → Free as many trades as they want only last transaction written to chain

Indirect Payment Channel → 3rd Person charlie involved

no direct channel between Alice and Bob, but channels between Bob and Charlie & Charlie and Alice exist

→ Charlie acts as the middle man



This can be extended to the Lightning Network where Alice can send something to Eric via already open channels

This is cheaper and faster than opening a new channel each time. Steps in the middle get a fee.

09) Algorithms for Fully distributed Systems

Bittorent is a peer to peer file shaving protocol.

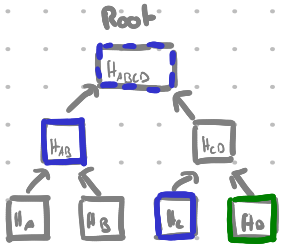
- breaks files into small pieces, users both down- and upload

Core components: - Tracker (Vermittler) coordinates peers and has list of active users
- Torrent File: Metadata about files and tracker

BitTorrent is network-friendly because more users = more bandwith → scales nicely
↳ uses technologies like merkle proof, Bloom filters, DHT Distributed hash tables

Merkle Trees (also used in Blockchains)

- Binary Hash Tree constructed bottom up
- To proof a transaction is in a block, only a few hashes are needed.



To be verified

☐ needed to compute Root Hash

- This scales super well $\log_2 4$. For 1 mio hashes only 20 needed to verify root. 2^{20}

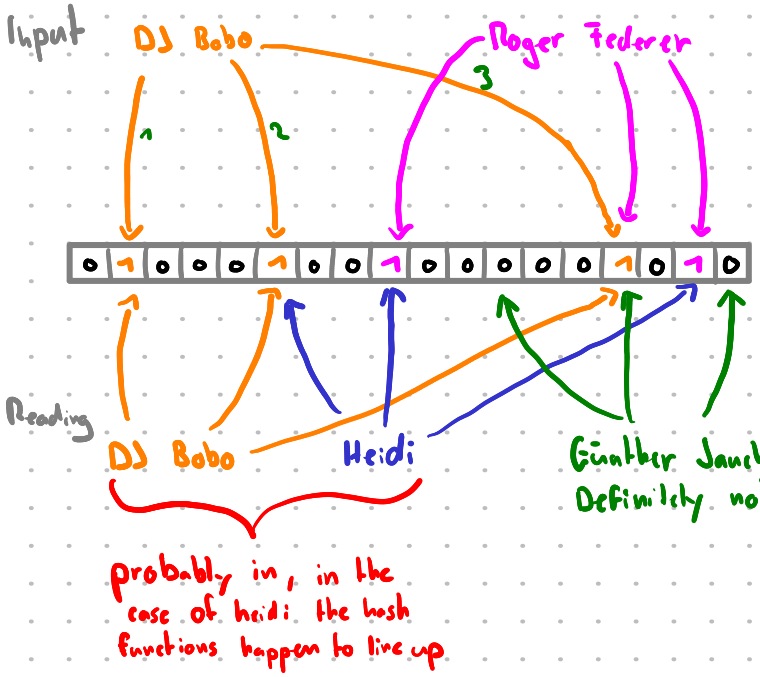
Magnet links: URI scheme that does not point to a centralized tracker ? WTF s.C

Bloom Filter

Bloom Filters are spare efficient compared to things like a Hash Table.

Drawback: · It can definitely say an element is not in → no false negative
· It can only say that an element is most likely in → probalistic

Removing elements is impossible. / Multiple Hash functions are used



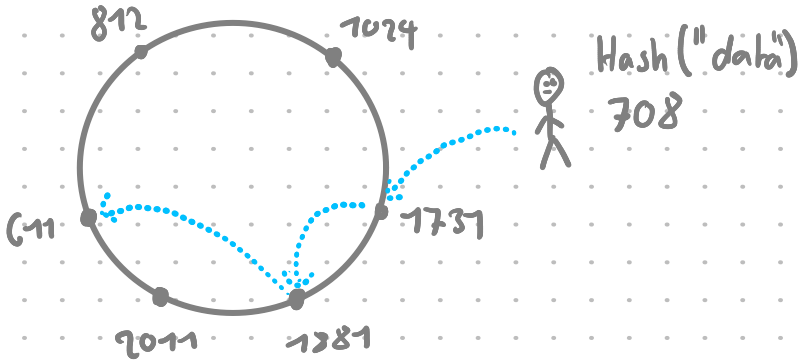
· all bits set to zero at first
· sets bit to 1 after hash function

· Bloom filter is optimal until about 50% filled
· used for fast search in LinkedIn
· Distributed systems use it for cache optimization (according to a yt vid)

Distributed Hash Table (DHT)

In peer-to-peer systems the main challenge is where to store data and how it can be found
↳ needs to scale well and be robust against errors
↳ (other than DHT, options would be central server or Flooding search (Bitcoin))

Structured Indexing: scalability → node only stores $O(\log(N))$ hops → route will be found after max $O(\log(N))$ hops.



Fundamentals for DHT:

· Desired characteristics: Reliability and Scalability
· Equal distribution of content among nodes → important but also difficult.
· Assignment of new nodes, remove old ones → requires remapping of DHT

- Routing to data item → big steps in the beginning then smaller steps once closer
  ↳ node knows many close nodes and fewer nodes far away

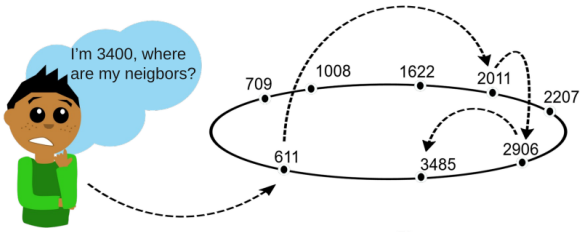
Join/Leave

- Joining of a new node
  1) Calculation of node ID (normally random / or based on PK)
  2) New node contacts DHT via arbitrary node (bootstrap node)
  3) Lookup of its node ID (routing)
  4) Copying of K/V-pairs of hash range (in case of replication)
  5) Notify neighbors
- Failure of a node
  - Use of redundant K/V pairs (if a node fails)
  - Use of redundant / alternative routing paths
  - Key-value usually still retrievable if at least one copy remains

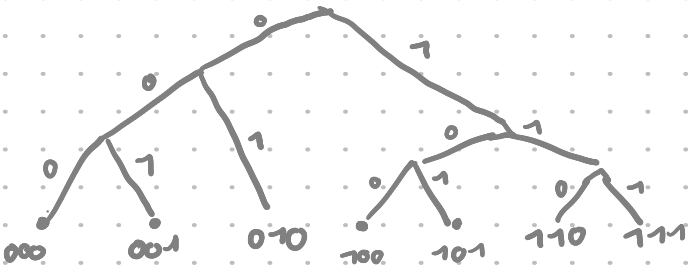
- Departure of a node

not sure if this is important, can't be asked to write this myself

  - Copying of K/V pairs to corresponding nodes
    - Can be before or after unbinding
  - Friendly unbinding from routing environment
    - If unbinding is unfriendly, need for keep-alive messages



Kademlia is one of the ways of building a DHT. Example



#6 looks for 3 6 xor 3 $110_b \oplus 011_b = 101_b = 5$

- 2^3, max size 8, #6 searches for 3

1	2	3
7	4 (or 5)	0 (or 1, 2)

Routing Table of #6
6 xor 3 = 101b → in 3 bits dargestellt → deshalb wird in bucket drei gesucht

- Neighbors of 6, if k=1

1	2	3
1	2	4 (or 5, 6, 7)

Routing Table of #0
0 xor 3 = 11b

- Search for 3, ask 0, neighbors of 0
- Ask 2, neighbors of 2

$000_b \oplus 011_b = 11_b$ → 2 bits → bucket 2

1	2	3
-	0 (or 1)	4 (or 5, 6, 7)

Routing Table of #2
2 xor 3 = 1b
distance 1 is the shortest

- Ask 2, 2 replies 0. 6 figures that there is no closer node, 2 is the closest one (2 xor 3 =1)

Sybil attacks: create a large number of nodes (more than honest) and take them offline all of a sudden. Data will disappear

↳ Prevention: creating all those nodes costs money

Redundancy of data: - Replication → originator is responsible for the data and periodically refreshes it. If he gone, data gone

Indirect Replication → closest peer is responsible, versioning more difficult and requires cooperation of peer and originator.

Replication makes consistency in a distributed system worse as it requires coordination.

10) Holochain

"Think local, act Global" → act autonomously while contributing to the collective.

Cryptographic data integrity without global consensus.

Characteristics Holochain:

- local validation of cryptographic integrity
- Each agent has its own source chain plus a DHT
  - shared data space across network
  - used for validation by network peers
- No native token required
  - signed entries and actions
  - immutable history of all activities

⚠ Achtung, hier wird Mut zur Lücke angewendet weil ennig kompliziert und quasi keine Use-cases im Real life

10) SUI

Layer 1 Blockchain for scalability with a focus on high throughput and low latency.

a Block is confirmed in less than a second with low transaction fee

Used for DeFi, gaming, NFT

↳ multiple commands in a single transactions

Unique architecture:

- Object centric model: all digital objects have an ID enabling independet processing
- parallel processing because of no-conflict transaction resulting in more Speeeeeed

Has a native SUI token

Validator network does proof of stake with ~100 validators atm

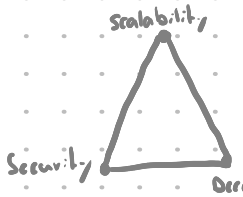
11) Ethereum Layer two solution

Layer 1 = Base chain → main Ethereum Blockchain

Layer 2 = additional networks/protocols on top of that. Transactions processed externally then written back to Mainnet.

↳ 10x - 100x lower fees, more transaction per second

Blockchain Trilemma



Difference to sidechains: ·store their data independently

sidechains don't inherit security from mainnet

2 types of L2 solution / rollups:

- Optimistic Rollup: assume that all transactions are correct with time to challenge them (↦ 7 days) with a fraud proof
  - Otimism or Arbitrum

- Zero Knowledge Rollup: mathematically verified transaction correctness
  - uses Zero Knowledge proof
  - No waiting time because transaction instantly valid once proofed

Key components for L1/L2 communication

On-chain: Smart Contracts

Off-chain:
- provers and validator for verification
- State management to post state to L1

Flow of a transaction L2 (in Arbitrum)

1) User starts transaction on Arbitrum
2) Sequencer (not distributed) confirms → L2 security level
3) Each ~ 15 min, transactions stored in batch written to Ethereum → L1 sec. lvl
4) After 7 days → Full security

Sequencer (run by OfficeChain)

- No SPOF as users can hand in transactions via L1 themselves
- Anyone can run an Arbitrum node

From L1 to L2

1) ETH locked in smart contract
2) Message to L2 to mint WETH
3) auto unwrapped to ETH on L2

From L2 to L1

1) initiate withdrawal
2) wait 7 days
3) claim on L1

Transaction fees on both to be payed