Decentralized Systems

Ethereum (cnt)

Consensus in Ethereum

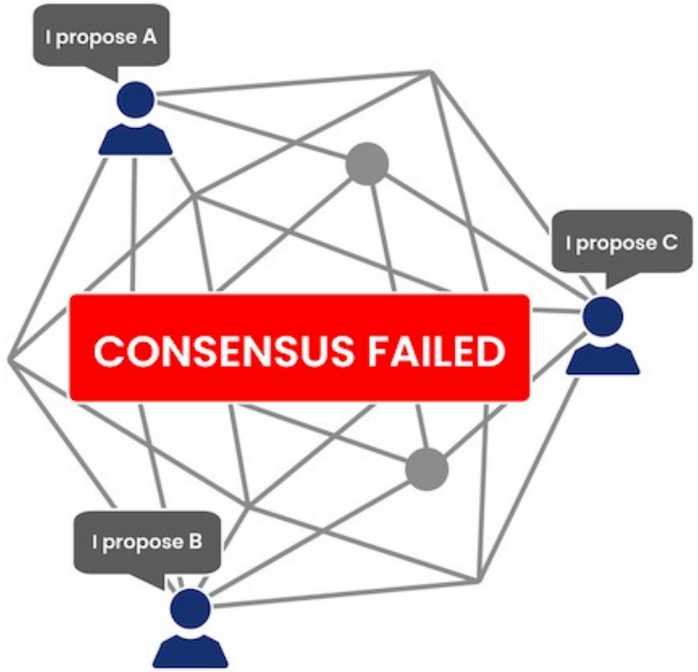
Consensus algorithms

1982 — Byzantine Generals Problem

2009 — Bitcoin PoW

2015 — Ethereum PoW

2022 — Ethereum PoS



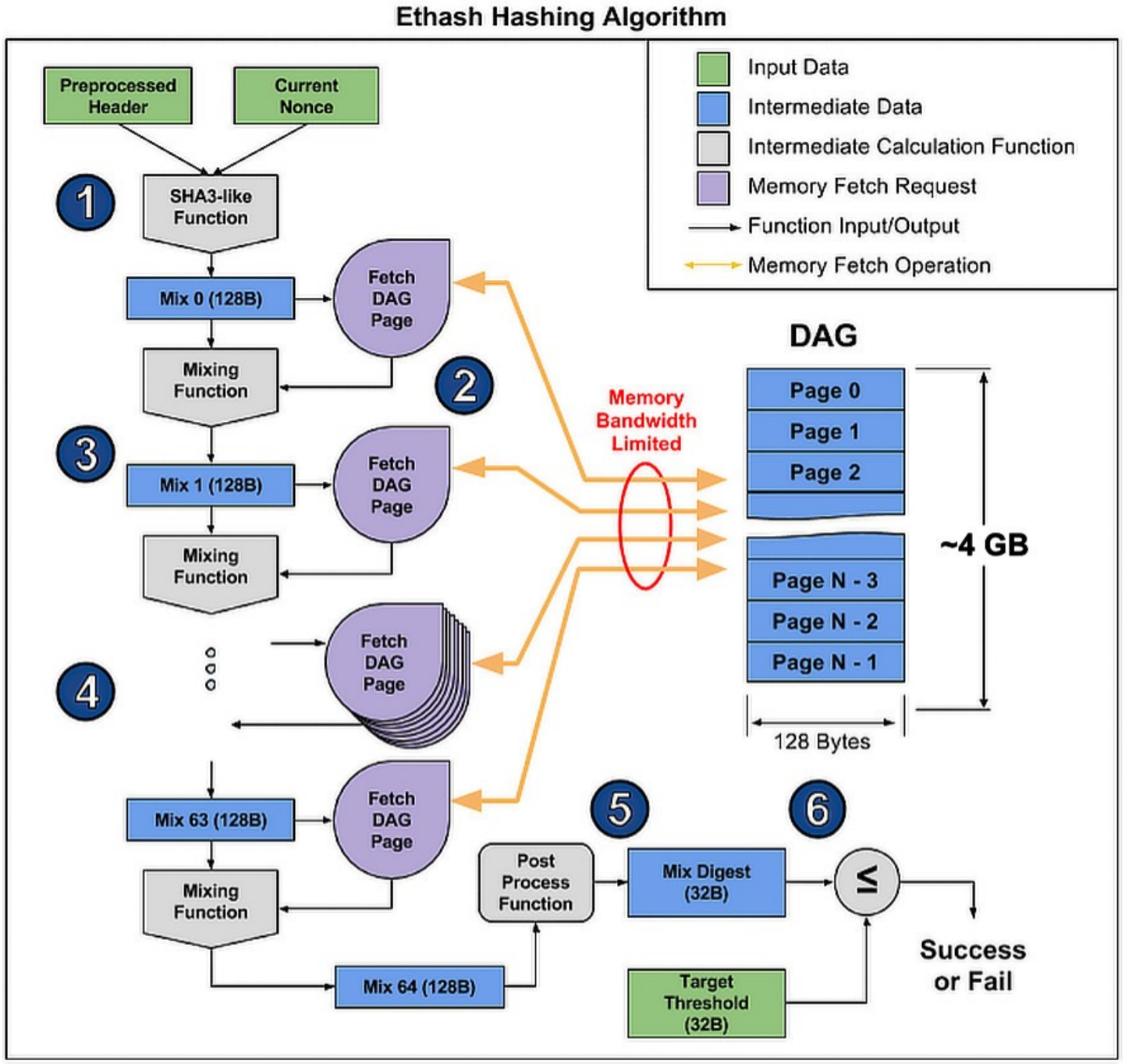
Past: Proof-of-Work

- **Ethash** is the PoW algorithm **previously used** in Ethereum
- **Memory-hard** algorithm, it requires a large amount of memory to mine efficiently (ASIC resistant)
  - miners must download and store a **randomly generated dataset** known as **DAG** (Directed Acyclic Graph)
  - a **new DAG** file of increasing file is generated every **mining epoch** (**30000 blocks**) , and miners must always use the most recent version
  - see https://minerstat.com/dag-size-calculator

Past: Proof-of-Work

- The difficulty of the PoW dynamically adjusts such that, on average, **one block** is produced by the network **every 12-15 seconds**
  - Orphan blocks, mined but not included on the main chain
  - Transactions included in an orphan block will not be finalized until they are included in a block on the main chain
- Miners compute the **mix hash** (Keccak-256) to reach the target and win the block reward
- Also in this case, difficult to compute, easy to verify

Past

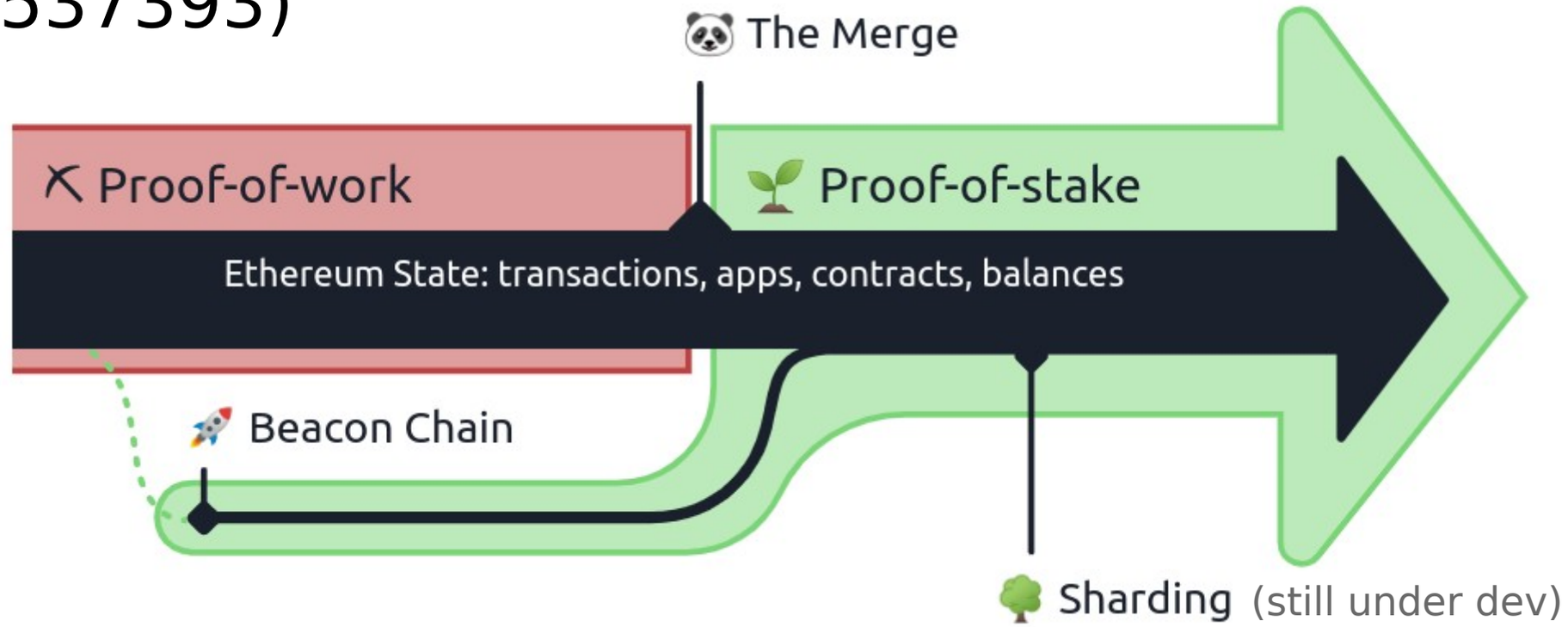

Ethash Hashing Algorithm
Input Data
Intermediate Data
Intermediate Calculation Function
Memory Fetch Request
Function Input/Output
Memory Fetch Operation
Preprocessed Header
Current Nonce
1
SHA3-like Function
Mix 0 (128B)
Fetch DAG Page
Mixing Function
2
3
Mix 1 (128B)
Fetch DAG Page
Mixing Function
4
Fetch DAG Page
Mix 63 (128B)
Fetch DAG Page
Mixing Function
Mix 64 (128B)
Memory Bandwidth Limited
DAG
Page 0
Page 1
Page 2
Page N - 3
Page N - 2
Page N - 1
~4 GB
128 Bytes
5
Post Process Function
6
Mix Digest (32B)
≤
Target Threshold (32B)
Success or Fail

Past: Proof-of-Work

- Steps for the miners
  - select a random starting point (page) in the DAG
  - follow a path through the DAG, visiting each node (page) only once
  - compute the Keccak-256 hash of the node's data and the mix hash from the previous step
- Compare the final mix hash with the target

Now: Proof-of-Stake

- Sept 15, 2022: **The Merge** (block number 15537393)



https://ethereum.org/en/upgrades/merge/
https://www.youtube.com/watch?v=EEuPmA8w0Kc

Now: Proof-of-Stake

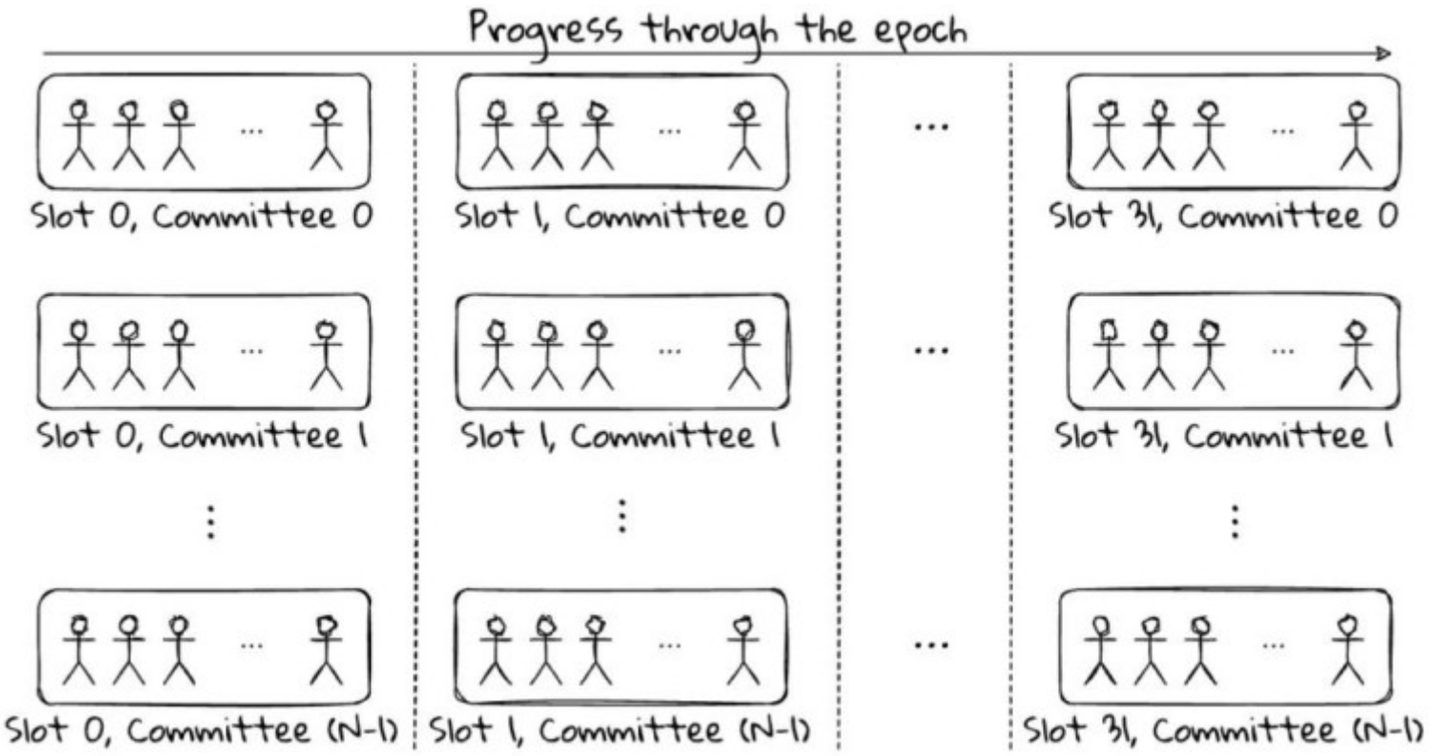
- To participate in PoS validation, users must **stake at least 32 ETH**, **individually** or through a **staking pool**
  - staked ETH is locked up and validators must be online to participate in the consensus process
- **Validators** are **randomly selected** to propose and attest new blocks
- If a validator proposes an **invalid block** or attempts to **double-spend** their ETH, they are **penalized**: some locked coins are burned! This is called **slashing**

Now: Proof-of-Stake

- The algorithm works in **epochs of 32 slots** (12 sec per slot)
- For each epoch, **32 committee** are formed, their **128 members** being randomly selected from the pool of all active validators using a pseudo-random function
  - this ensures that the committees changes dynamically from epoch to epoch, preventing centralization of power within the network
- Committees
  - **elect their proposers** responsible of the **next block** in their slot
  - **vote to validate their block** (attestation)

Now: Proof-of-Stake

- Proposers and validators are rewarded for their job



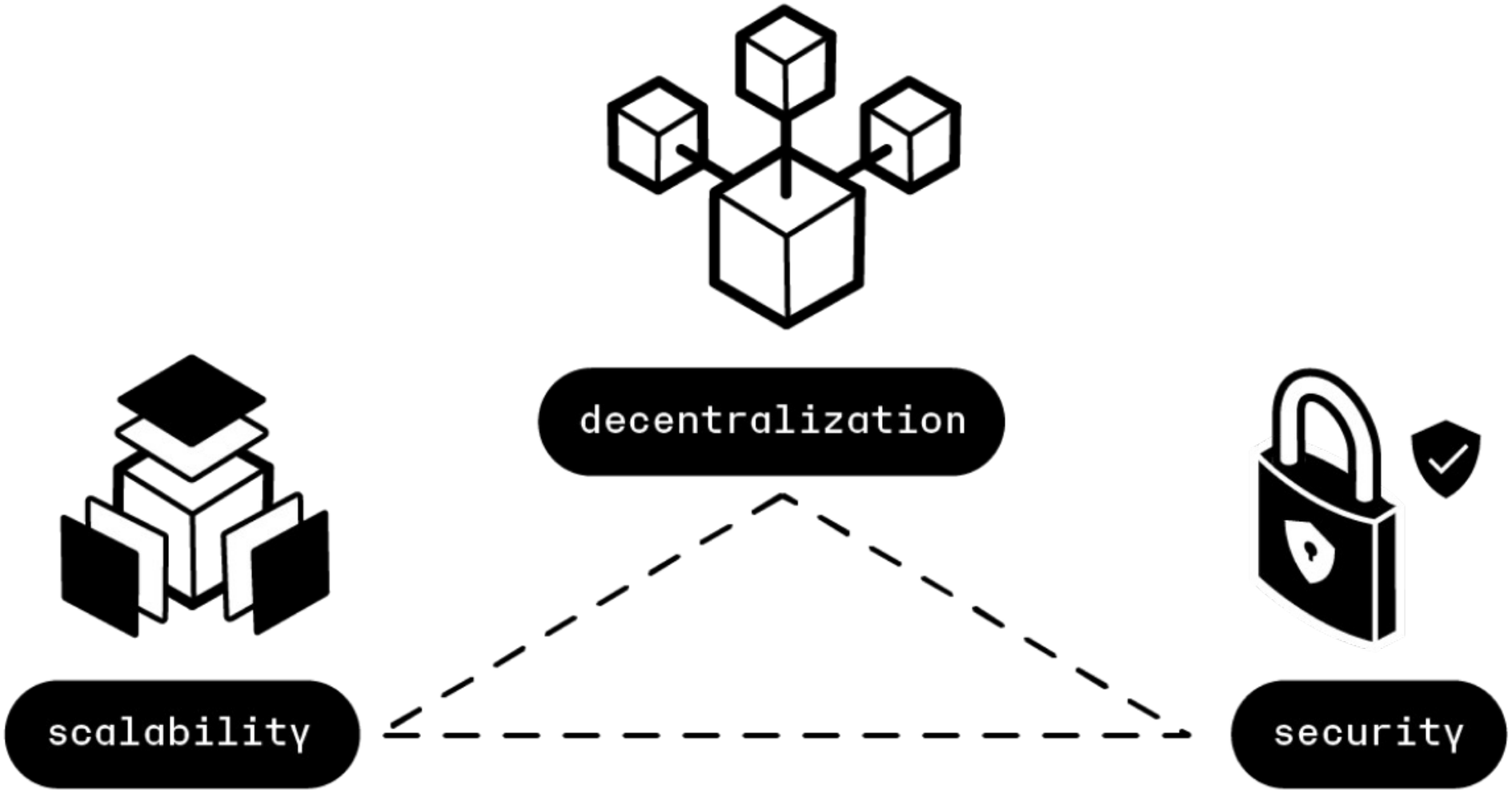
PoW  vs PoS 

- **Gas fees will remain the same** because the Merge does not affect the execution layer where the fees are determined
- **Transaction speed might slightly improve** but will not dramatically increase (a new block every 12 sec)
- **Malicious validators are slashed**, e.g., a significant part of their stake can be burned, up to the whole stake of 32 ETH in the worst case
- **Note: staked ETH can now be withdrawn** (after Shanghai upgrade, April 12, 2023, block 16112764)

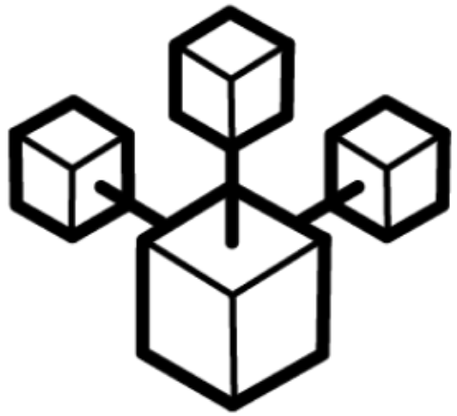
Blockchain Trilemma

- Ethereum improved by 99% (not sure!) in terms of energy consumption after The Merge, and set the stage for future upgrades that will make it faster and more scalable (sharding)
- Attempt to tackle the Blockchain (or Scalability) Trilemma

Blockchain Trilemma

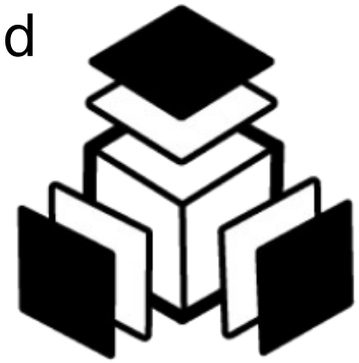


Blockchain Trilemma



degree to which a blockchain network is **controlled** by a single entity or group of entities

ability to **process** a large number of transactions per second



decentralization

scalability

security

ability to **resist attacks** from malicious actors

Tokens

Tokens

- **Tokens can represent anything** physical or digital goods, e.g., a ticket for a concert, a collectible, points at the supermarket, casino coins, etc.
- In Ethereum they
  - are managed by the underlying blockchain
  - can be issued with a **few lines of code**
  - are accessible with a (often dedicated) wallet
  - https://etherscan.io/tokens

Tokens

- Can be implemented on different layers of the technology
  - **Protocol tokens**
  - intrinsic, native, or built-in tokens, have the role to **keep the network safe** from attacks (mining rewards) and for **preventing transactions spam** (transaction fees)
  - BTC, ETH

Tokens

- Can be implemented on different layers of the technology
  - **Second-layer tokens**
  - these are application tokens which **can be created via a smart contract** (Ethereum) or issued in a Layer 2 network

Tokens

- Think of Ethereum like the internet and all the dApps as websites that run in it
- dApps are decentralized, not owned by a single entity, they are owned by people
- This usually happens by a crowd-sale called the **ICO** (Initial Coin Offer) or the **ITO** (Initial Token Offer)

Tokens

- ICOs have changed the way projects are funded
  - Assemble your team and create a product
  - Create the Tokens to be sold during the ICO
  - Write the White Paper
  - Run a well designed website and social channels
  - ...

Ethereum tokens

- Various standards
  - **ERC-20** for fungible tokens (ERC-223, ERC-777)
  - **ERC-721** for non fungible tokens (NFTs)
  - **ERC-1551** supports non-fungible and fungible tokens

ERC = Ethereum Request for Comments

ERC-20

- Defined in https://eips.ethereum.org/EIPS/eip-20
  - They have a property that makes **each token be exactly the same** (in type and value) **of another token**
  - They have functionalities to transfer tokens from one account to another, to get the current token balance of an account, to delegate other accounts to spend tokens
  - See https://www.openzeppelin.com/

ERC-721, known as NFT

- Defined in https://eips.ethereum.org/EIPS/eip-721
  - A non-fungible token (NFT) is a **special type** of cryptographic token which **represents something unique**
  - NFT have varying properties, and **represent scarce assets** like art, collectibles or real estate
  - Can also represent **identities**, **certificates** (licenses, degrees), voting rights, medical data, etc.

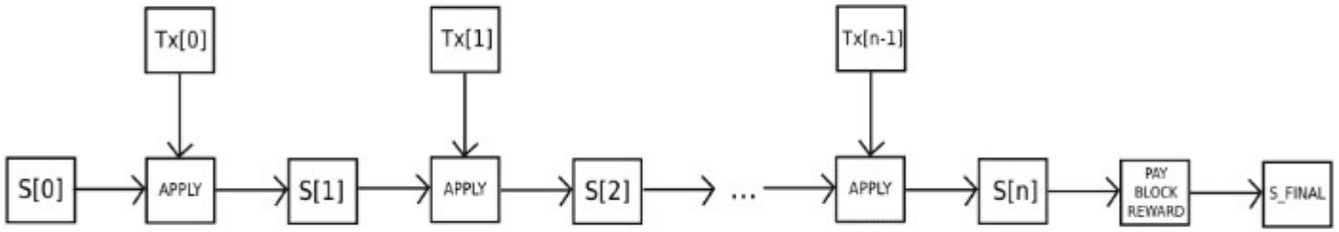
ERC-721, known as NFT

- See for example
  - https://www.cryptokitties.co/
  - https://www.larvalabs.com/cryptopunks
  - https://opensea.io/
  - https://www.coingecko.com/research/publications/10-most-expensive-nfts-ever-sold

Data Structures

State machine

- Ethereum uses the idea of the **world state** which is stored as a mapping between account addresses and account states (balance, contract code, contract storage)



Source: https://ethereum.org/en/whitepaper/

State machine

- Users send **transactions** Tx[i] which force nodes in the network to **change state** S[i+1]
  - Owned → Owned: transfer ETH between users
  - Owned → Address 0: deploy contract
  - Owned → Contract: call contract with ETH & data
  - Contract → Owned: contract sends funds to user
  - Contract → Contract: one program calls another (can send funds)

State machine

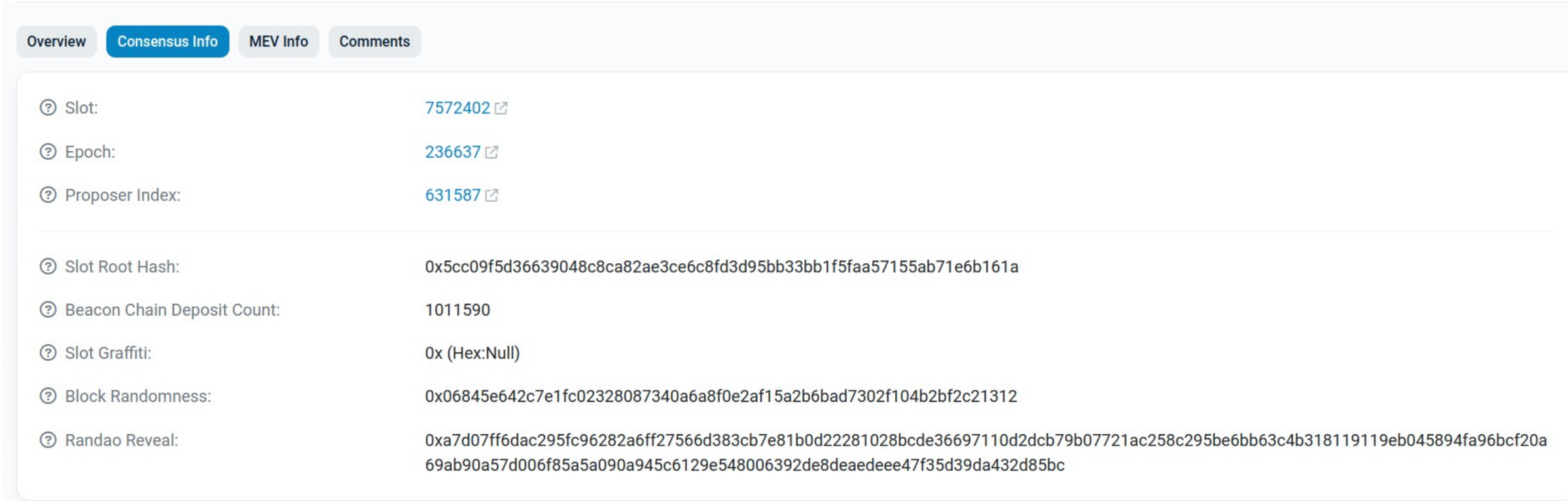
- Validators collect transactions Tx[0]...Tx[n-1] from users and the Proposer creates a new block
- To produce a block the Proposer
  - for i=0,...,n-1: execute (APPLY) state change of Tx[i] sequentially (can change state of > n accounts)
  - record updated world state in the block
- Other validators re-execute all Txs to verify the block and, if valid, sign it; enough signatures → epoch is finalized

World state data structure

- The world state is stored in a optimized tree structure known as **Merkle Patricia Trie**
  - **P**ractical **A**lgorithm **T**o **R**etrieve **I**nformation **C**oded **I**n **A**lphanumeric
  - Tree-like data structure in which each node represents a prefix of a string. The root node represents the empty string, and the child nodes represent the strings that start with the prefix of the parent node
  - Plus hashes... (Merkle tree)

Block header data (simplified)

(1) Consensus Info: proposer ID, parent hash, etc.

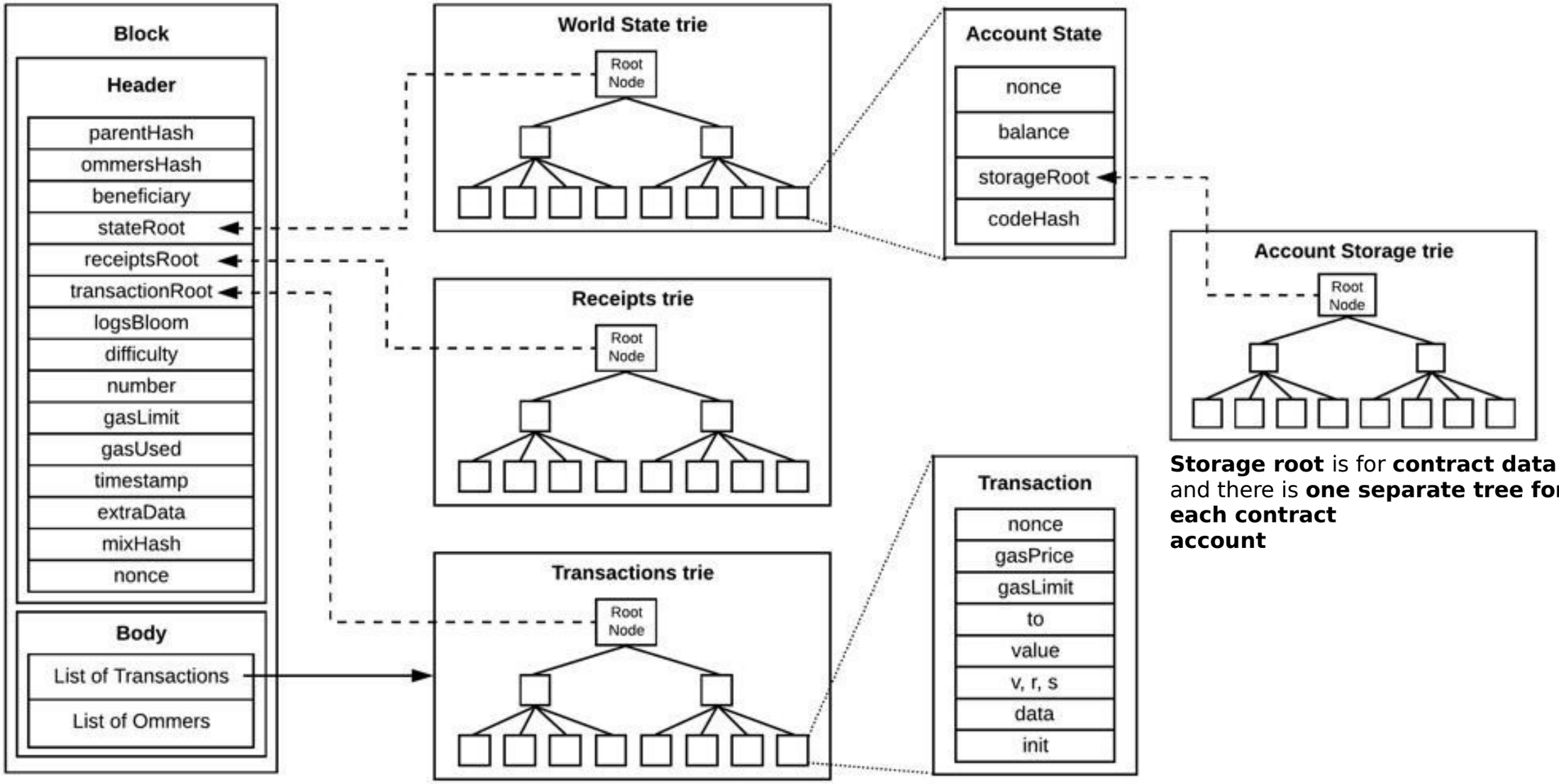


Block header data (simplified)

(2) Address of gas beneficiary: where fees will go
(3) Gas used: used to adjust gas price

(4) Root hashes

Hash:	0x4e9e72e833fcff968366c861b7592a1ba6a03abb37c29b87858b0b66c002b872
Parent Hash:	0xd47e03df946d265666365e8f97eec3d0ac2fb1bf9a3a41fe6bb50d4a82249e05
StateRoot:	0xc9ca9a12de6aae94c8a09a311e069165cebdc8892a1603cf3b6d802478174e46
WithdrawalsRoot:	0x47a3a48871623455710b325f5d261da31b3be2e44e9b29cc6b1b713904adf2fe
Nonce:	0x0000000000000000



Storage root is for **contract data** and there is **one separate tree for each contract account**

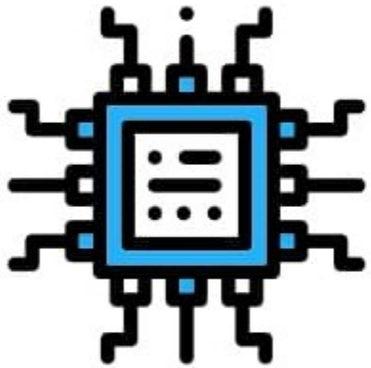
Smart Contract

Smart Contract

1

Smart Contracts are **written as code** and committed to the blockchain. The code and conditions in the contract are **publicly available** on the ledger.

2



When an event outlined in the contract is triggered, like an expiration date or an asset's target price is reached-- the **code executes**.

3

Regulators can watch contract activity on the blockchain to **understand the market** while still **maintaining the privacy** of individual actors.

Smart Contract

- 1993: Nick Szabo defined **smart contracts** as a set of **promises**, specified in digital form, including **protocols** that facilitate, verify, or enforce how parties perform these promises
  - A set of promises (IF...THEN...ELSE)
  - Digital form
  - Protocols for communication and performance
  - Self executing program

Smart Contract

- Everyday example
  - the vending machine

- A modern example
  - installment payment of a car: a smart contract is able to check if the user has regularly payed the installment
  - if the check is negative then the smart contract renders the car inoperable

Smart Contract

Turing complete

- Any instruction can run
- Maximum freedom
- Easiest to make mistakes

Restricted Instructions

- Limited instructions
- Purpose-specific actions
- Hard to make mistake
- Harder to hack

Off-Chain execution

- Contract code is not on blockchain
- Code distribution has been coordinated
- Hard to make mistakes

On-Chain execution

- Contract code is stored on blockchain
- Code changes are part of blockchain consensus
- Very transparent

Smart Contract

- Ethereum is the **largest ecosystem of decentralized applications** or dApps built on a **decentralized network**
- Smart contracts on Ethereum are **public** and **transparent** – like open APIs – so dApps can also reuse code written by others
- Smart contracts are as good as the developers building them: “buggy” smart contracts stay in the blockchain forever

Web2 vs Web3 (bard)

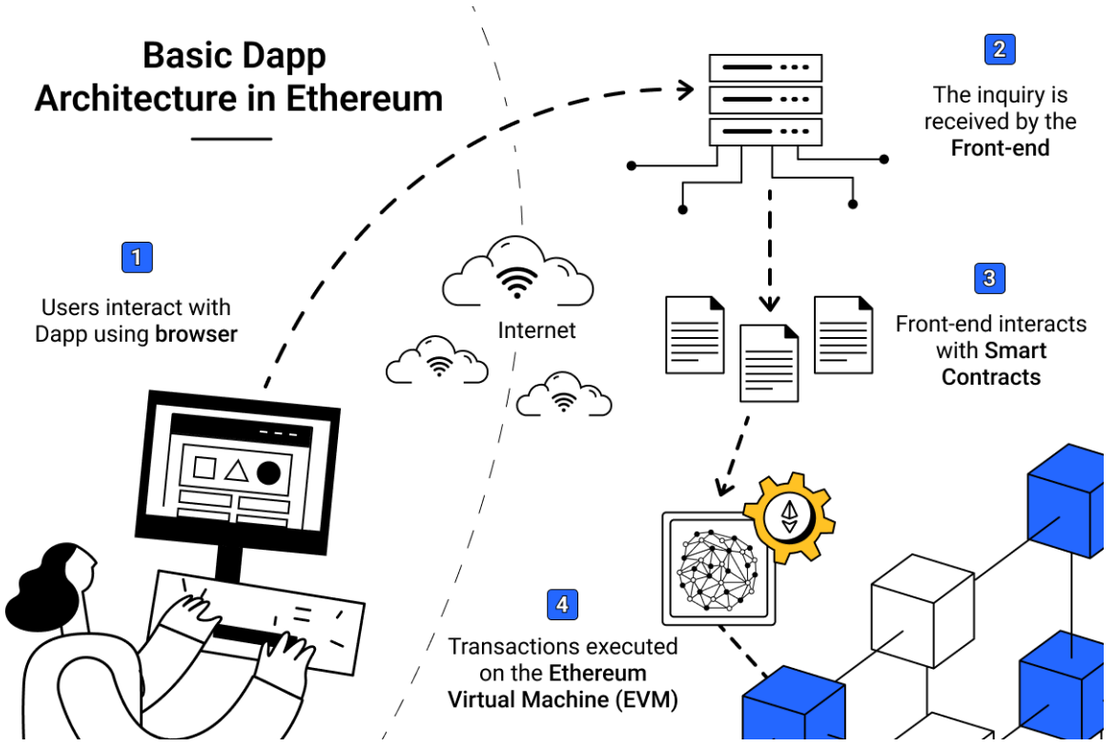
Feature	Web 2 app	Web 3 dApp
Ownership	Centralized	Decentralized
Data storage	Centralized	Decentralized
Control	Controlled by a single entity	Controlled by a network of computers
Security	Vulnerable to hacks and data breaches	More secure and resistant to censorship
Transparency	Not transparent	Transparent and verifiable
Accessibility	More accessible	Less accessible, requires some technical knowledge

Popular dApps (bard)

- **DeFi (Decentralized Finance)**
  - Uniswap
  - Aave
  - MakerDAO
  - Compound
  - SushiSwap
  - Curve Finance
  - PancakeSwap
- **Gaming**
  - Axie Infinity
  - Decentraland
  - Gods Unchained
  - Alien Worlds
  - Dark Country
  - My Crypto Heroes
  - Chainmonsters
- **NFTs (Non-Fungible Tokens)**
  - OpenSea
  - Rarible
  - LooksRare
  - SuperRare
  - Axie Marketplace
  - Magic Eden
  - SolanaArt
- **Other**
  - Brave Browser
  - Filecoin
  - Storj
  - Golem
  - Audius
  - Livepeer
  - Ocean Protocol

Activity

- Please select one dApp each, google 5 minutes and tell the class what it is about



https://info.etherscan.com/decentralized-applications-an-overview/