Introduction to blockchain

Bitcoin: Trustless, Permissionless, Censorship resistant, No double spending

Smart contracts: a computerized transaction protocol that executes the terms of a contract

Tokens: are smart contracts that contain the logic for storing and updating balances of token holders

Properties of DeFi: Non-custodial, Permissionless, Openly auditable, Composable

Bitcoin

Blockchain/Block: data structures that store information, such as transaction data

Blockchain: p2p; Data is recorded in multiple identical data stores (ledgers) that are collectively maintained by a distributed network of computers (nodes); Consensus algorithm

Block: block header + txn_count + txns

- block header: <version><previous_block header_hash><merkle_root_hash><time><...> <nonce>
- txn_count: total number of transactions
- txns: every transaction in the block

Hash functions:

- **Pre-image resistance**: For hash value h in output space, hard to find x s.t. H(x) = h.
- **Second pre-image resistance**: For x & H, hard to find different y s.t. H(y) = H(x).
- **Collision resistance**: For H, hard to find different x & y, s.t. H(x) = H(y).

Merkle tree: encode a set of values into a single hash

- Log time inclusion proof generation & time/storage inclusion proof verification

Proof of Work (PoW): a leader election process in which participating nodes (miners) invest computational power in solving cryptographically hard, memoryless puzzles

- Partial pre-image attack on SHA256
- Probability of a hash being a solution: p = Target / 2^{256} Expected number of hashes = 1/p = 2^{256}/Target
- Solve-time follows a Poisson distribution (avg. 10 min)
- Difficulty algorithms adjust the highest possible target

Mining: Miners are nodes that try to solve the PoW puzzle

- incentivized to select transactions because of Transaction fees & Block reward
- Finite supply of 21 million BTC; Reward halves ≈ 4 yrs

Longest chain rule: accepted as the correct history

- **Block tree**: all valid blocks whose previous linked blocks are known
- **Active chain**: a single path from the genesis block to a leaf node of the block tree (every path is a valid path)

Soft fork: Protocol changes that remain backward compatible with older protocol versions

Hard fork: Protocol changes which can incur a permanent split of the blockchain (not backward compatible) – blocks considered invalid under previous protocol rules

- **Bitcoin Cash**: block size 1 MB -> 32 MB, receive equal BTC

Transactions: Data structures that encodes the transfer of funds from an input source to an output destination

- Created → propagated → validated → added to the ledger of transactions
- no confidential info, 300 - 400 bytes of data
- <version><input counter><inputs><output counter> <outputs><lock time>

Unspent Transaction Output (UTXO): Fundamental building block of a transaction; Chunks of BTC locked to some address

- Transactions consume one or more UTXOs
- UTXOs must be consumed entirely
- Sending BTC: creating a UTXO that belongs to recipient
- **Transaction inputs**: UTXOs consumed + Unlocking scripts that meet spending conditions
- **Transaction outputs**: UTXOs created by a transaction
- **Transaction fees**: inputs - outputs

Scripting: Stack-based, conditional jump, not Turing-complete

- Transaction is invalid if script fails

Redeem scripts: to enable scripting with UTXO

- Only a hash of the script is published beforehand; full scripts are published when spending
- Write script -> Hash script to create address -> Receive Bitcoins -> Publish script and required data (usually signature) using a transaction
- Lack persistent state, Turing-completeness, transparency

Ethereum

Similarities w/ Bitcoin: data structure, forks, PoW until recently

Ethereum Classic: does not revert TheDAO hack

Differences w/ Bitcoin: PoS, block rewards, accounts, EVM

Proof-of-Stake (PoS): an alternative consensus mechanism

- Validators "stake" some ETH to participate
- Each epoch, a validator is randomly selected to create a new block; Other validators verify & attest
- If a validator produces an invalid block, ETH stake slashed

Pros of PoS: energy efficient, low barrier to entry (32 ETH), more decentralized, requires less participation incentive

Cons of PoS: not as battle tested, complex, larger attack surface

Validator rewards: receive rewards for staking & participating; proportional to % of total staked ETH

Reward types: source vote (checkpoint), target vote (checkpoint), head vote (block), proposer reward, block fees

Penalties: Validators are penalized when they do not vote in a timely manner for the amount they should have received; NOT penalized if they miss a head vote or are late to propose a block

Slashing & forcefully removed: propose & sign 2 blocks for same slot; Attest to source/target blocks that surrounds another one; double vote 2 candidates for same block

Finality: block cannot be changed anyone without burning significant funds

- PoW: probabilistic finality, final after n blocks
- PoS: first block of each epoch is a "checkpoint block"
- Two checkpoint blocks have a "supermajority link" if they are both attested by at least 66% of the total ETH staked
- A transaction is finalized if it is included between two checkpoint blocks that have a supermajority link

Accounts: a mapping between addresses and account state

Externally owned accounts: created by generating private/public key pair; address is derived from the public key; can initiate transactions

Contract accounts: deployed as smart contracts & controlled by their code; no associated private key; can't initiate transactions

Ethereum account info: nonce (txn_count), balance (1 ETH = 1e18 wei), codeHash (n/a for EOA), storageRoot (has of storage as Merkle tree)

Smart contracts: Programs deployed on a blockchain

- written in high-level language & compiled into bytecode
- Interacted with using transactions

EVM: Stack-based VM; no functions, only jumps, no types; has regular VM instrs & instrs to interact with environment; ephemeral and permanent storage; 256-bits words

Throughput: txns/sec = no. of txns per block / block time

Block time: no. of sec between 2 blocks; BTC: 7 tps, ETH: 25 tps

- higher the throughput, the harder for validators to participate & more centralized

Alternative blockchains:

- **Solana**: speed, PoS, PoHistory, centralized↑, outages↑
- **Cardano**: first PoS, slow development
- **TRON**: delegated PoS, throughput↑, fee↓, centralized↑

Ethereum overview: run decentralized programs on EVM, blockchain as storage, network of nodes stores exact copies, non-custodial, permissionless, completely transparent

- **Blocks**: **Header** (slot, proposer index, parent hash, state hash, gas limit, gas used, timestamp, base fee per gas, transactions root), **Body** (Info on state & executed txns), **Accounts**, **Transactions** (nonce, priorityFee, gasLimit, to, value, signature, data, init), **Other states**
  - World state σ_t -> σ_{t+1} determined by all transactions included in a block
- **Nodes**: EVM, Mempool; **User**: Transactions

Transaction: cryptographically signed instruction sent by EOA

- Only Transactions Cause State Change by transfer of a balance or execution of smart contract code

Consensus: agree on state, transition to next, ensure validity

- Bitcoin uses the "longest chain rule"
- Ethereum uses the "heaviest chain rule", highest no. of accumulated validator votes weighted by staked balances

Ether: native cryptocurrency of Ethereum; balances directly stored as world state; used to pay transaction fees (gas)

Tokens: implemented & stored in smart contracts; ERC20, anyone can crate

EVM objectives: execute code in a P2P network in a deterministic way; verifiability; atomicity

EVM operation: nodes in network runs EVM impl. on own machines; rerun code executed by proposer to verify; code is in storage & validity can be verified using the code hash

Deploying Code: complied into bytecode & deployed via a transaction containing code to create a new contract addr; hash of code stored in world state

Executing Code: node includes txns w/ calls to contract -> verity validity -> execute code locally -> broadcast results

Gas: A measure of computational effort; computed on opcodes

- prevent DoS, infinite loops, reward validators/miners

Base Fee: set by protocol based on the previous block; burnt

Priority Fee: This fee is set by the person submitting a transaction as a tip; goes to the miner of the block

Mempool: a set of pending transactions

- Transactions are submitted as an RPC request to a node
- choose transactions for inclusion in order of profitability

Smart Contracts: A set of promises, specified in digital form, including protocols within which the parties perform on the other promises; Programs deployed on a blockchain

- Can: Turing complete, persist data, transfer money
- Can't: interact w/ outside, scheduled periodically

Solidity: Strongly typed, simple type system, multi-inheritance

- Write -> test -> optimize for gas -> compile -> deploy
- Global objects: block, gasleft, tx, msg
- Value: bool, int/uint, address, byte arrays, enum/set
- Reference: structs, arrays, mappings
- if, else, while, do, for, break, continue, return, revert
- **view**: read-only; **pure**: no read, no write, return value generated solely from function arguments
- **external**: real EVM message call, can't be called internally
- **internal**: only accessible from within the contract
- **public**: internal + auto-generate getter functions
- **private**: internal but not visible in derived contracts

memory: ephemeral. Lifetime limited to external function call

storage: lifetime same as contract lifetime; needs more gas

calldata: non-modifiable, ephemeral for function arguments

Inter-contract Communication: message calls

- Every transaction wrapped in a message call

Smart Contract Development

Interface: contains the function signatures of a contract, but not implementation; used to define the contract's public API library contains function definitions that can be used by other contracts.
dynamic linking: w/ public functions, deployed as a contract
static linking: only internal functions, embedded
Hardhat: JS/TS; **Brownie**: Python; **Foundry**: Solidity
ERC/EIP: Ethereum Request for Comment
ERC-20: a standard for fungible tokens; represent currencies, shares; can check balances, transfer tokens, approve transfer
Fixed point: cannot represent fractions of tokens
ERC-721: a standard for non-fungible tokens; represent unique; digital asset; associated URI to e.g. IPFS

Oracles

Oracle Problem: Smart Contracts are unable to connect with external systems, data feeds, APIs, existing payment systems or any other off-chain resources on their own.

- Centralized Oracles are a Point of Failure

DeFi

DeFi support: smart contracts must support inter-contract communication, be expressive enough to encode financial protocol rules, allow conditional execution and bounded iteration, feature atomic transactions so that no execution can result in an invalid state
Application Binary Interface: standard way to interact with contracts; ABI & address enough to interact w/ contract; JSON
Composability:
New Contract Instances: safest
Known Interface & Existing Instance: need right address & type
Existing Instances and Low Level Calls: 'blind'
DeFi: a peer-to-peer powered financial system; Non-custodial (full control over fund), Permissionless (w/o being censored/blocked), Openly auditable, Composable, (pseudo) anonymous, new capital efficiencies
Keepers: external agents who can trigger state updates.
Oracles: a mechanism for importing off-chain data into the blockchain virtual machine
Governance: the process through which an on-chain system is able to change the terms of interaction

Protocols for Loanable Funds (PLFs)

PLF: a protocol which establishes a distributed ledger-based market for loanable funds; demand from borrowers, supply by savers, interest rate; Allows to trustlessly borrow & save, earning interest; way to access other funds; On-chain leverage
Overcollateralized: a borrower needs to post collateral which is worth more than the value of the loan; for asset with additional utility / income streams or shorting an asset
deposits A -> deposits B -> takes out A against B & pay interest
Short: Supply asset A and borrow asset B; Sell asset B and buy asset A; If price of asset B falls, buy back asset B and repay loan
Long: supply A, borrow B, sell B for A, hope A appreciates
Liquidations: When a loan's overcollateralization ratio falls to a specified threshold, liquidators can repay the outstanding borrowed amount on behalf of the borrower by purchasing part of the collateral at a discount
Close factor: upper bound for a single liquidator to repay
Liquidation threshold: % at which becomes undercollateralized
Health factor = $\sum$(Collateral in ETH) × L_t / Total Borrows in ETH

- H falls below 1, the loan can get liquidated

Liquidation penalty: the discount for collateral purchase
PLF Components: actor, controller, markets, oracle, liquidators
Utilization: Total % borrowed out; no idle liquidity at 100%; decides interest

Flash loans

Flash loan: an undercollateralized loan without a borrowing limit that must be repaid in the same transaction borrowed

- Fee is charged on the borrow amount
- If loan is not paid, revert atomicity

Arbitrage: takes loan of B, buy A w/ B, sell A to get B elsewhere
Collateral swap w/o repay loan: takes loan of B & deposit, withdraws A, swap A for B, repay B
borrow() -> execute() -> repay -> execute() end -> borrow() end
Yield aggregators: logic for allocating funds yield-maximizingly

- Deposit 1 asset & aggregator allocates to ≥1 protocols
- External keepers monitor when the interest rate

Automated Market Makers (AMM)

Real-world exchanges: Order-book based; market price is in between the lowest ask and the highest bid; centralized crypto

- Issues: expensive, trust and governance

AMM: most common way to buy and trade ETH and tokens

- Liquidity Provider for A & B, Liquidity Pool, Trader

Pricing Mechanism: Invariants for calculating market price
Constant Sum: $k = B_\alpha + B_\beta$; impractical unless 2 assets equal
Constant Product: $k = B_\alpha \times B_\beta$; $(B_\alpha + d_\alpha)(B_\beta + d_\beta) = B_\alpha \times B_\beta$; $d_\beta = -(d_\alpha B_\beta) / (d_\alpha + B_\alpha)$; price α in terms of β = gradient; e.g. Uniswap
Liquidity Providers: trading fees, liquidity mining (governance)
Problems of AMM: large portion of liquidity only available at the extremes of the pricing curve; slippage for larger trades
Stableswap: combines constant product & constant sum invariant; can choose custom liquidity ranges; flattens curve
Impermanent Loss: LPs with less valuable assets b/c arbitrage
Imbalancing LP: manipulate price to attack contracts relying on AMM; exacerbated by single-asset liquidity provision
Borrow A on flash loan -> swap A for B on AMM -> borrow A on contract -> swap B for A on AMM -> repays A
Pros of AMM: no order book maintenance, simple implementation of CP AMM, low gas fees
Cons of AMM: can have impermanent loss/coin de-peg, high slippage for low liquidity market, sandwich attacks to users
Ground truth: Off-chain ground truth, On-chain ground truth, On-chain estimates of off-chain ground truth
Oracles: mechanism for importing off-chain data to blockchain
Centralized Oracles: requires trust in the data provider
Decentralized Oracles: rely on incentives for accurate and honest reporting of off-chain data, e.g. Consolidated Price Feed

Stablecoins

Stablecoin: A cryptocurrency with an additional economic structure to stabilize price and purchasing power
Custodial: trust in a third party; Reserve Fund, Central Bank

- takes custody of asset X and then issues on-chain tokens

Non-custodial stablecoins: do not require trust in a third party; Source of Value from store of value/collateral in the stablecoin; **implicit collateral** (supply dynamically adjusted to stabilize price), **endogenous collateral** (asset created for the stablecoin), **exogenous collateral** (collateral has use outside of stablecoin)
Risk Absorbers: speculators who absorb system risk
Stablecoin Holders: making up demand
Issuance: mechanism for creating stablecoin units
Governance: mechanism for deciding system parameters
Data feed: importing off-chain data - via an oracle
MakerDAO's DAI: overcollateralized debt of 170%; Price of ETH can fall significantly w/o collateral being worth less than debt
Mechanisms for 1:1: System of individual vaults, interest rates, Auction mechanisms, Peg Stability Module allows swapping
Liquidation mechanism: collateral auction (Dutch); "Dog"
Debt Auction: MKR auctioned off for an amount of DAI
Risk Dimensions: Deleveraging spirals, Oracle risk, Governance risk, Blockchain/contract risks, Censorship/counterparty risk

Technical and Economic Security

Technical exploits: Smart contract vulnerabilities, Single transaction exploits, Ordering exploits; risk-free
Technical security: secure from an attacker who is limited to atomic actions (e.g., not possible to steal assets)
Logical bugs: authorization, lack of invariants, lack of checks, Integer manipulation
Compound bug: Reentrancy; attacker contract makes reentrant call to vulnerable contract, which reads stale state
Defenses: Testing (Unit/integration), Fuzz testing, Formal verification, Security audits, Bug bounties
Transaction Ordering: attacks may involve front- and/or back-running within a single block, thereby undermining the technical security of DeFi protocols
Front running: Taking profitable actions based on (typically non-public) information on upcoming trades in a market

- B submits copy of transaction with higher priority fee

Miner/Maximal Extractable Value (MEV): The value a miner/validator can extract from deciding tx order & inclusion

- Miner includes a copy of same transaction

Sources: DeExchanges -> arbitrage, liquidation mechanisms
Miners/Validators can exclude, determine order, insert own
Anyone can front/back-run
Single Transaction Attacks: can be executed in a single tx; success of attack doesn't depend on order of transactions
AMM manipulation: temporarily imbalance an AMM; e.g. Harvest Finance Exploit
Governance attack: obtain sufficient governance token to propose & execute malicious contract code; steal funds
Transaction Ordering Attacks: success depends on order
Displacement attack: front runs some target transaction; does not depends on whether the target transaction is executed
Sandwich attack: alters the deterministic price on an AMM prior to and after some other target transaction has been executed in order to profit from temporary imbalances in the AMM's liquidity reserves
A swap X for Y -> B swap X for Y w/ more fee -> B -> A -> B sell Y
Issues w/ EV from tx ordering: bidding wars for which tx(s) get included result in gas price volatility, failed txs still get included and take up block space unnecessarily (they revert)
Defense: private mempool & comm. channels, e.g. Flashbots
Economic security: not profitable for an attacker who can perform non-atomic actions to manipulate the protocol

- non-atomic, not risk-free, need models of markets

e.g. protocol uses time-weighted average AMM price as oracle

Scalability and Bridges

Types of node: full (DB copies: N, block: all), archival (DB copies: all, block: all), pruned (DB copies: N, block: N)
Fork rate = (stale_block / blocks)*100%
block propagation: time to reach others, freq. of new block
Big block: fewer forks; **small block**: lots of forks
Scalability: Reduce fork rate & maximize population of nodes
Data availability layer & settlement layer: Ethereum
Execution layer: Off-chain system
Bridges: hold
ed assets on ETH & checks liabilities off-chain

Open Challenges

Privacy preserving (e.g., zero-knowledge proofs, multi-party computation) computationally expensive, e.g. ZCash, Monero
Decentralization: geographically concentrated Ethereum nodes
Composability risk: imbalancing AMMs & seizing governance
Extractable Value: MEV, Governance EV, Oracle EV