BLOCKCHAIN

Store of value: An asset that maintain its value over time and will retain its purchasing power in future, like gold

Blockchain: A Database of smart contracts Blockchain is intentionally walled off and it can't connect to real world or read and listen to data

Smart Contract: are set of instructions executed in a decentralized way without need of centralize or third party intermediary. Also called trust minimized agreements as they are tamper-proof and don't need for trust in any central authority.

Ethereum:- a blockchain protocol

Ethereum Virtual Machine:- EVM is a runtime environment for smart contracts, where each contract is run into its own virtual sandbox environment.

Ethereum vs Bitcoin:- Bitcoin is just store of value but Ethereum protocol is both store of value (ETH) and utility to facilitate these decentralized agreements (smart contracts).

the Oracle:- Oracle is any device/third party service that interacts with off-chain world to provide external data or computation to smart contracts.

Hybrid smart contracts:- on-chain + off-chain agreements.

Chainlink: A modular decentralized oracle network that can both bring external data and external computation into our smart contracts.

DAPP: Decentralized Application = Decentralized protocol = smart contracts.

DAOs: Decentralized Autonomys Organizations.
Gas: A unit of computational measurement: The more complex your transaction is, the more gas you pay.

Hash: A unique fixed length string, meant to ind identify a piece of data. They are created by placing said data into a "hash function."
Genesis Block: First block in blockchain
Hash Algorithm: A function that computes data into a unique hash.
Mining:- The process of finding the ~~problem~~ "solution" to the blockchain "problem".
Block: (A combination of block, nonce, transaction and previous hash to create a unique hash for this block.) :- A list of transactions mined together.
Nonce: A "number used once" to find the "solution" to the blockchain "problem."

Signing
Private key: only known to key holder/owner
Public key: is used to sign transactions.
Both Eth and Btc use Elliptic Curve Digital Signature Algorithm (ECDSA) to create public key.
Ethereum address is a piece of your public key. (last 20 bytes). ~~It can~~ this method to derive address can change blockchain to blockchain or even in different versions of ethereum.

Gas II: Block Rewards & EIP1559

On Ethereum, Sending Ethereum takes 21000 gas
Base fee:- The minimum gas price to send your transaction
Block time: Time How long it takes between blocks being published.

Block reward :- The amount of cryptocurrency awarded to miners for being first to solve a complex math problem and creating a new block of verified blockchain transactions. Every time a new block is created it is verified by all other competing miners.

EIP :- Ethereum improvement proposal.

Old ~~current~~ fee model (Depreciated now) It was based on first-price auction method. Users who want to have their transaction picked up by a miner had to essentially bid for their space in a block. They would pay a gas price for a transaction. Miners were incentivized to pick up transactions with highest gas price. This would be quite inefficient as users who pay high fees would have their transactions executed first and they would end up overpaying.

EIP 1559 goals:-

- Making transaction fees more predictable.
- Reducing delays in transaction confirmations.
- ~~Using~~ Improving user experience by automating fee biding system.
- Creating a positive feedback loop between network activity and Eth supply.

EIP 1559 Fee Model:-

It introduced a base fee. and increase in network capacity achieved by changing the max gas limit per block from 12.5M to 25M

EIP 1559 logic:

If network is more than 50% utilization, base fee is incremented and when it is lower than 50%, base fee is decremented. So network aims to achieve equilibrium at 50% by adjusting fees.

It also introduced a miner tip, a seperate fee that can be paid directly to miner to incentivize them to prioritize a transaction

Nonce(again):- It works like OTP. short for "number used once". It is a random or incremental number that is included in transaction in order to prevent replay attacks. It is typically a 64-bit integer and included in transaction data.

Replay attacks occur when an attacker intercepts a valid transaction and then broadcasts it again to network causing the recipient to receive same payment twice. By including nonce in each transaction, blockchain network can verify that transaction is unique and has not been previously executed. A nonce cannot be used twice.

Gas:- A unit for measuring the amount of computation required to perform a certain task on Ethereum blockchain.

Fee: To calculate fee ⇒ Multiply gas cost by gas price.

~~Gas price~~

Gwei: Gas price measuring unit.

1 Gwei = 0.000000001 Eth

Mempool:- The place where all pending transactions wait for miners to pick them and include them in next ethereum block.

Node:- A single instance in a deceneralized network.

Consensus: Consensus is the mechanism used to agree on the state of a blockchain. Roughly a consensus protocol of in a blockchain can be broken down into two pieces.

1) Chain Selection 2) Sybil Resistance.

Sybil Resistance:- a mechanism that is designed to prevents sybil attack is a decentralized system.

A sybil attack is an attack in which an attacker creates multiple fake identities or nodes to gain control over a network.

Sybil resistance mechanisms:- Two types
i) Proof of work ii) proof of stake.

PoW:- No matter how many pseudo anonymous accounts an attacker can make, each one still has to undergo this very computationally expensive activity of finding the answer to the proof of work problem/riddle, which is finding a nonce (that basically work as an OTP in a transaction) This riddle depends on blockchain and can vary accordingly. With proof of work, it's a verifiable way to figure out who the block author is and be sybil resistant. Now you need to combine this with a chain selection rule to create this consensus.

Chain selection rule:- How do we know which blockchain is the real and the true blockchain. Bitcoin and Ethereum, both uses NAKAMOTO consensus. which is combination of PoW and longest chain rule. the decentralized network decides that whichever blockchain ~~that~~ has the longest chain or most number of blocks on it is going to be the chain they use.

Block confirmation:- Nuber of additional blocks added on after our transaction went through in a block.

PoW is not consensus protocols but a part of it. But sometimes people use it interchangeably. In PoW, all nodes competes against each other to solve blockchain riddle. and which ever wins gets the transaction fee. This process uses a lot of energy as all nodes virtually on network are trying to solve it first. They also get block reward directly from blockchain itself. Blockchain cut block rewards in half roughly every four years.

Some blockchain, like Bitcoin, have a set time when they're no longer going to give block reward and miners or nodes only get transaction fee.

Attacks :- There are two types of attacks that can happen in blockchains.

i) Sybil attack ii) 51% attack. (control blockchain many to

defined in previous page. (a single entity pretends to be

ii) 51% attack :- As part of our consensus protocol, blockchains are going to agree that the longest chain is the one they are going to go with, so long as it matches up with 51% of the rest of the network. If you have longest chain and have 51% of rest of the network, you can fork in the network and bring the network onto your now longest chain. (which essentially creates a new version of blockchain)

Longest chain rule :- Whichever blockchain has most buy in and is the longest is the blockchain that the whole system is going to corroborate. When nodes produce a new block and add to the longest chain, other nodes will follow this longest chain that the rest of the network is agreeing with, add those blocks to their chain and follow up. In simple terms, longest chain acts as elder of the house. All other members do or agree with elder. So if someone has influence or control over elder, they can manipulate whole blockchain (51%) attack

drawbacks of PoW :- Uses a lot of energy.

PoS :- Proof of stake nodes put up collateral as a sybil resistance mechanism. If they misbehave, some of the stake will be slashed or taken away.

this is a very good sybil resistance mechanism because if you create whole bunch of anonymous accounts, then each of those must have some cryptotoken (Btc, Eth) as stake and if you misbehave, you will end up by losing all the money you put as collateral. In this system, miners are called validators because they're no longer binding anything. They're just validing other nodes. Unlike PoW where each node race against eachother, PoS nodes are randomly chosen to propose the new block and rest of validators will validate if that node has proposed the block honestly.

Randomness:- Blockchains are deterministic systems. They are walled gardens from the rest of the world. And deterministic system by definition can't have random numbers. Chosing random validators and actually choosing the node will change from blockchain to blockchain, but Eth 2.0 is using "Randao" or "Randomness beacon"

Deterministic system:- A system in which output or outcome can be predicted with certainity based on a given set of inputs or initial conditions. In other words, a system which produce same result every time it is run with same input or some starting condition

Pros of PoS:- • great sybil resistance mechanism
• way less computationally expensive to figure out new block (use less energy)

Cons of PoS:- • Considered a slightly less decentralized network, due to upfront staking cost to participate. which gets into philosophical bottle about how decentralized is decentralized enough. But the general consesus among blockchain engineers

is that proof of stake is very decentralized

Scalability:- Gas prices can get really high if a lot of people want to send a transaction because a block has limited block space (previously 12.5M now 25M). This limits scalability, because as more people add to blockchain, it cost more and more to use blockchain. So there's a ceiling to how many people can use system at a time because of financial constraints. Ethereum 2.0 is also tackling this by new methodology called sharding.

Sharding:- A sharded blockchain really just means that it's going to be a blockchain of blockchains. There is a main chain that coordinates everything amongst several chains that hook into this main chain. This means there are more chains for people to make transactions on effectively increasing amount of block space that there is. Sharding can greatly increase number of transaction on a blockchain layer one.

Layer 1:- Base layer blockchain implementation Bitcoin and Ethereum is layer 1.

Layer 2:- Any application built/added on top of layer 1. (blockchain). eg. chainlik, arbitrum or optimisum.

Arbitrum and optimisum are known as Rollups. and they rollup their transactions into a layer one like Ethereum.

Rollups:- Rollups are kind of like a sharded chain. they derive their security from base layer (from layer one) like ethereum and they bulk send their transactions onto layer one. they help increase throughput and reduce transaction

fees by compressing multiple transactions into a single transaction that is recorded on the blockchain (layer 1). Now these rollups are different from sidechains. Sidechains derive their security from their own protocols, while rollups derive their security from the base layer. Base layer is also called mainnet.

Sidechains:- Sidechains are seperate blockchains that are connected to main blockchain. they allow for faster and more efficient transactions processing by offloading some transaction processing to sidechains.

Sidechains and rollups are both layer 2 scaling solutions for blockchain networks, but they work in different ways and have different pros and cons.

Rollups are generally considered more secure than sidechains because they rely on security of main blockchain. However, they can be more expensive to implement and may require complex development. Sidechain, on the other hand, are generally less expensive and easier to implement, but they require additional security measures.

Solidity basics.

Pragma:- short for pragmatic information is used to provide additional instruction to compiler or preprocessor like compiler version.

Data Types:-

Boleon:- True, false

Uints:- Unsigned integer (isn't +ve or -ve but represent +ve whole numbers.

ints:-

string:-

address:- Ethereum address.

bytes:-

visibility:-

Public:- Public variable implicitly get assigned a function that returns its value (creates getter func for storage/state variables).

Private:-

Internal:-

External:-

arbitrary:-

View and pure:- functions, when called alone, don't spend gas and they disallow modification of state. Pure functions also disallow reading from state. If there's a function that is updating state that calls a view or a pure function that is the only time it'll cost gas. i.e. if a gas calling function calls a view or pure function- only then it will cost gas.

Storage vs State:-

Storage:- refers to persistent, on-chain storage of contract's data.

State:- refers to current value of a contract's data at a particular time.

Data locations

EVM can access and store information in six places.

i) Stack ii) Memory iii) Storage
iv) Calldata v) Code vi) Logs

Storage:- Permanent persistance on blockchain.

Calldata:- Temporary input that is passed to a function as argument. it is read-only and can't be modified.

Memory:- Temporary data storage area that is available to function during execution. Its value can be changed.

Both memory and calldata types are erased/cleared once the function execution is completed.

We have to specify this only for array, struct or mapping types.

parameter in a function
if a variable is of type "uint", solidity automatically knows "uint" will live in memory. But arrays, structs and mapping types are special types in solidity. So we have to explicitly set a data storage types. If a "variable" in a global scope is defined as "uint", solidity will consider it as "storage" (permanent) type. A "string" type parameter in a function is actually an array of bytes behind the scenes.

Mapping:- Mapping is a data structure where a key is "mapped" to a single value. Kind of like objects in JS. syntax:-

declaration:- mapping (string => Uint256) public nameToNumber
(data structure, key type, value type, visibility/access type, mapping name)

initialization:- nameToNumber[_name] = _favoriteNumber
(mapping name, key name/variable, value name/variable)

In mapping, every key is initialized to ~~the value~~ to null value which for "Uint256" is "zero". If a 'uint' is in scope of a function and it's data location is not specified, it will be "memory" by default. But if "uint" is declared outside of any function, it is treated as state variable and stored in "storage" data location by default.

Library:-

A Library is a special type of contract that can be deployed on Ethereum blockchain and contains reuseable code that can be called by other contracts. You can't declare any state variables and can't send ether. A Library is embedded into contract if all library functions are internal. It can also be used to add more functionality to different values.

Chainlink:-

Chainlink Data feeds:- A network of chainlink nodes gets data from different exchanges and data providers and brings that data through a network of decentralized chainlink node (oracles), chainlink nodes use "weighted aggregation" median/% to figure out the actual price of asset and deliver that in a single transaction as a "reference contract", "price feed contract" or a "data contract" on chain that other smart contracts can use.

Trigger parameters:- A parameter that is used to initiate execution of a specific function within contract. there in data feeds, when a parameter is triggered, a function executes that will update the price of asset by getting new data off-chain.

Deviation threshold:- It is a value used to determine how much difference or variation is acceptable b/w a reported value and the actual value of a data point in a data feed.

Heartbeat:- is a regular message or signal sent by a device or system to indicate that it is still functioning properly. Heartbeat is typically sent after a regular intervals. In data feeds, it is a trigger parameter which is triggered after 3600 seconds/1hr to update price of asset.

arbitrary:-

Safe Math:- Before solidity 0.8.0, safe math was a big problem and a library named "Safe Math" was being used all the time. Problem was if you declare a 'uint8' variable which can only hold values from 0-255 and than added / increased this value from 255, it would be reset and would be set to the lowest possible value of that type. ~~like~~ for example:-

coder:
```
uint8 public number = 255.
function add() public {
    number = number + 1;
}
```

explanation: Before 0.8v of solidity, unsigned integers and integers ran on this concept of being "unchecked" which means if you pass the upper limit of a number it would just wrap around and start back from the lowest number it could be. In version 0.8, transaction fails if you try to pass upper limit. In this instance, transaction would fail & when we would try to add 1 into 'number' variable. 0.8v automatically check to make sure if you are going to "overflow" or "underflow" on a variable. We can actually revert back to unchecked version by using "unchecked" keyword. eg:

coder:
```
function add() public {
    unchecked { number = number + 1; }
}
```

ps: Now even 0.8v wouldn't fail transaction but it would revert number to 0.

Sending Ether

Transfer:- Sending ether from one address to another costs 21,000 gas. Transfer function is capped at 2300 gas. If more gas is used it throughs an error and revert transaction.

Send:- it is also capped at 2300 gas but if it fails it returs a boolean of whether or not it was successful. it would not revert transaction.

Call:- It is a lower level command and can be used to call virtualy any function in of etherium without even having to have the ABI. It doesnot have capped gas, so it forwards all gas or set gas and returns two variable. One is a boolean showing success or failure of call and the other stores returned data (if any) of the called function. *For the most part*, call is the recommended function to send or receive native token of blockchain.

Constructor:-

It is a function that is immediately called whenever a contract is deployed.

Modifier:-

A modifier is a reusable code that can be attached to a function defination to modify its behavior. They are commonly used to enforce certain conditions or requirements before a function can be executed.

Syntax:-

```
modifier onlyOwner {
    require(msg.sender == owner);
    _; // This indicates where the modified
}      // function code will be inserted
```

Advanced Solidity:

Immutable vs constants.

Constants:- are variables that cannot be modified. Their values are hard coded and can save gas cost. Syntax convention is to use uppercase letters. Like

// address public constant MY_ADDRESS = ...

This can't be changed after deploying ever.

Immutables:- are like constants and their values can be set inside the constructor but cannot be modified afterwards.

Syntax convention is to use uppercase or add 'i_',

e.g. public immutable MY_ADDRESS;

OR // public immutable i_member;

It also save gas like constants.

The reason that these two save gas is because instead of storing these variables inside of a storage slot, we actually store them directly into bytecode of the contract

Custom Errors-

Strings in errors cost a lot of gas to store even if they are smaller as each character is stored individually as a string array (as whole). As of 0.8.4V of solidity, the alternative is you can now use custom errors for reverts. 'require' cost much more gas than "revert". So we can use 'revert' with 'if' statements instead of require.

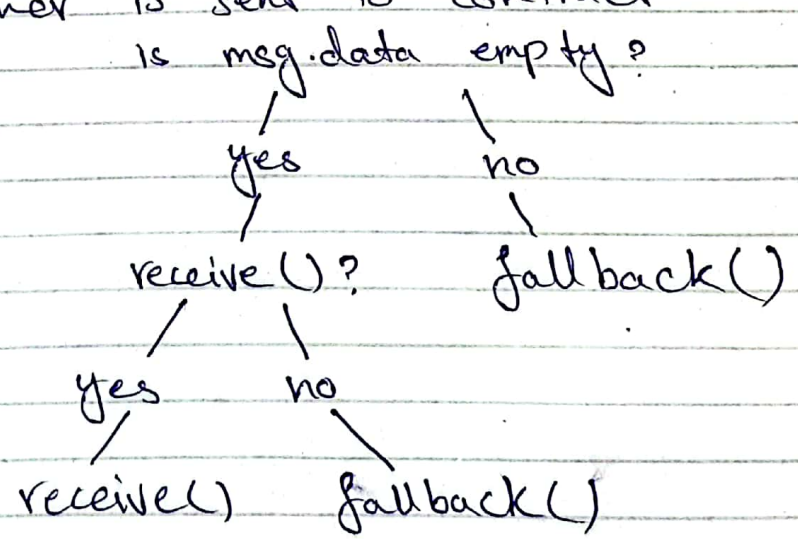
Receive & fallbacks.

These functions are special fuctions which are executed when

- a fuction is called that does not exist
- Ether is directly send to a contract

Receive:- A contract can have at most one receive function. This function cannot have any arguments, can't return anything and must have 'external' visibility and 'payable' state mutability. This is executed only on plain Ether transfers. If there's calldata 'fallback' will be triggered. Receive will be triggered anytime we send transaction and we don't specify function and we keep 'calldata' blank. given it is declared. It would be executed even if sent ether value is zero.

Fallback:- If a contract can have at most one fallback, similar restrictions like 'receive'. The fallback function is executed on a call to contract if none of the other function match the given function signature for/ if no data was supplied at all and there is no 'receive' function.



Blockchain theory
by Muhammad Muaz

Social links

Github: https://github.com/muazazhar
Linkedin: https://www.linkedin.com/in/mmuaz
FB: https://www.facebook.com/muaaz51