ETHEREUM FUNDAMENTALS

What is Ethereum Blockchain?

PEER TO PEER - IN DIGITAL CURRENCY WE TRUST - PEER TO PEER - IN DIGITAL CURRENCY WE TRUST

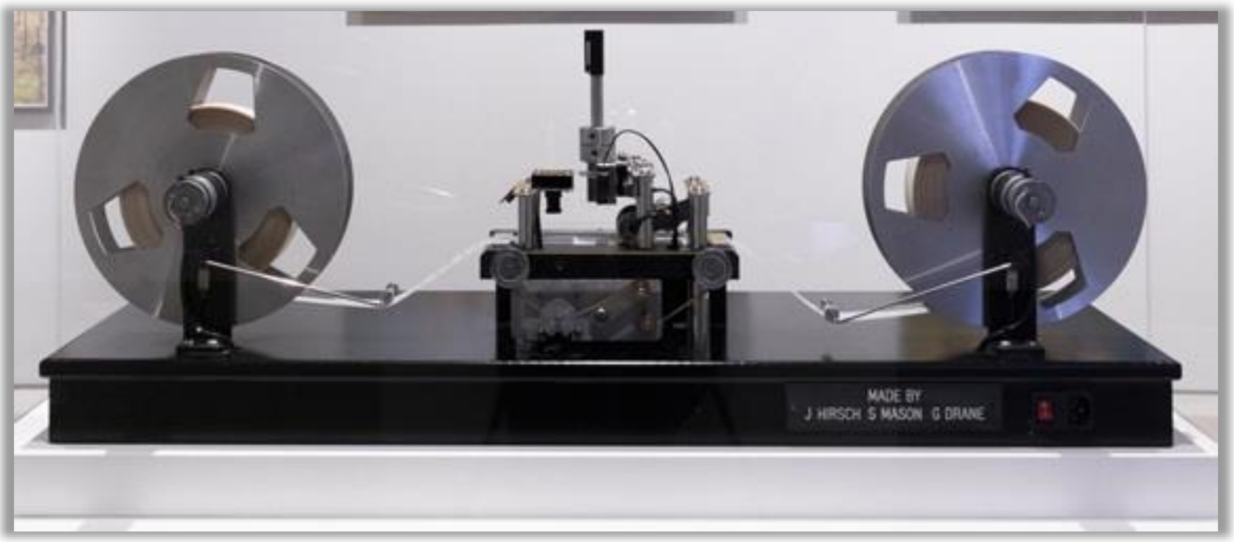
Ethereum is a decentralized blockchain platform that enables the development and execution of smart contracts and decentralized applications (DApps). It was proposed by Vitalik Buterin in 2013 and launched in 2015.

The Ethereum blockchain is a distributed ledger that records all transactions and smart contract interactions across its network of nodes.

At its core, Ethereum is a Turing-complete virtual machine called the Ethereum Virtual Machine (EVM), which allows developers to write and deploy smart contracts. Smart contracts are self-executing contracts with the terms of the agreement directly written into code.

Ether (ETH) is the native cryptocurrency of the Ethereum blockchain. It serves as a medium of exchange for value transfers within the network and also acts as a "fuel" to pay for computational operations performed by the EVM.

What is Turing Machine?

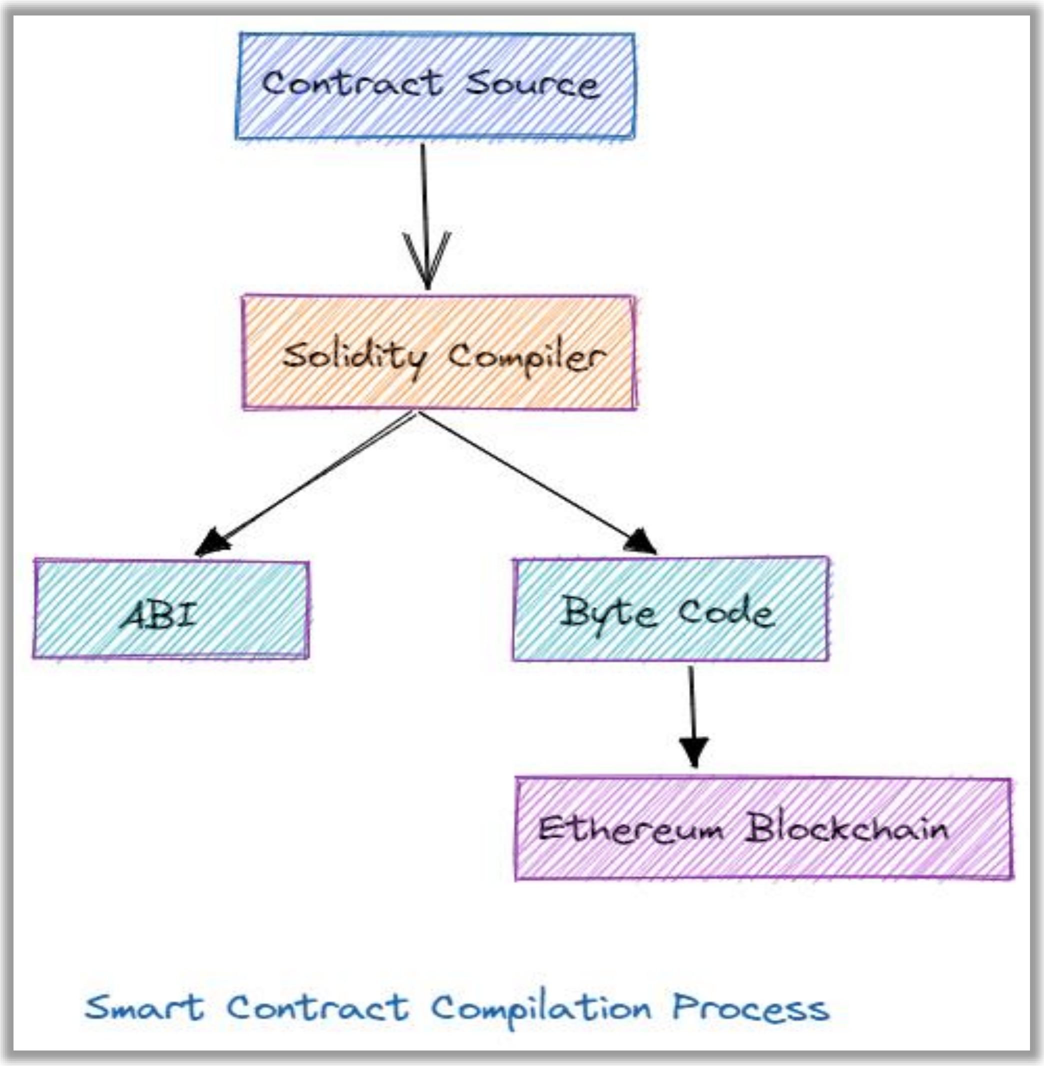


A Turing machine is an abstract mathematical model of a computing device that consists of an infinite tape divided into cells, a read/write head that can move along the tape, and a set of rules or instructions for manipulating the symbols on the tape.

It was proposed by Alan Turing in 1936 as a theoretical concept to understand the limits and capabilities of computational systems. The tape of a Turing machine is divided into discrete cells, each of which can store a symbol from a finite alphabet.

Turing machines serve as a foundation for understanding computability theory, complexity theory, and the theoretical underpinnings of modern computing.

What is Ethereum Virtual Machine?



The Ethereum Virtual Machine (EVM) is a key component of the Ethereum blockchain platform. It serves as a runtime environment for executing smart contracts, which are self-executing contracts with the terms of the agreement directly written into lines of code.

The EVM is a Turing-complete virtual machine, meaning it can perform any computation that a regular computer can, given enough time and resources.

Smart contracts written in high-level languages such as Solidity are compiled into bytecode, which is then executed by the EVM.

Smart Contract in Ethereum Blockchain:

In the Ethereum blockchain, a smart contract is a self-executing digital contract that is stored on the blockchain and automatically enforces the terms and conditions defined within it.

Smart contracts are written in high-level programming languages like Solidity and are deployed on the Ethereum network.

- **Decentralized Execution:** Smart contracts are executed on the Ethereum Virtual Machine (EVM), which runs on multiple nodes across the Ethereum network. This ensures that the execution of the contract is decentralized and not controlled by a single entity.

- **Immutable and Transparent:** Once deployed on the blockchain, smart contracts cannot be modified or tampered with. The code and the contract's state are publicly visible, providing transparency and immutability.

- **Self-Enforcing:** Smart contracts automatically execute actions and enforce predefined rules based on the conditions specified within the contract. They eliminate the need for intermediaries or trusted third parties to oversee contract execution, reducing the potential for fraud or manipulation.

- **Trustless Interactions:** Smart contracts enable trustless interactions between parties by removing the need for trust in a central authority. The execution of the contract is deterministic and can be verified by any participant on the network.

- **Integration with Cryptocurrency:** Smart contracts on the Ethereum blockchain can interact with Ethereum's native cryptocurrency, Ether (ETH). They can hold and transfer Ether, enabling complex financial transactions and decentralized applications (dApps) to be built on the Ethereum platform.

What is Solidity Programming Language?

Solidity is a high-level programming language specifically designed for writing smart contracts on the Ethereum platform. It is the most widely used language for developing decentralized applications (dApps) and executing smart contracts on the Ethereum blockchain.

Solidity is a contract-oriented language, meaning it allows developers to define and implement smart contracts.

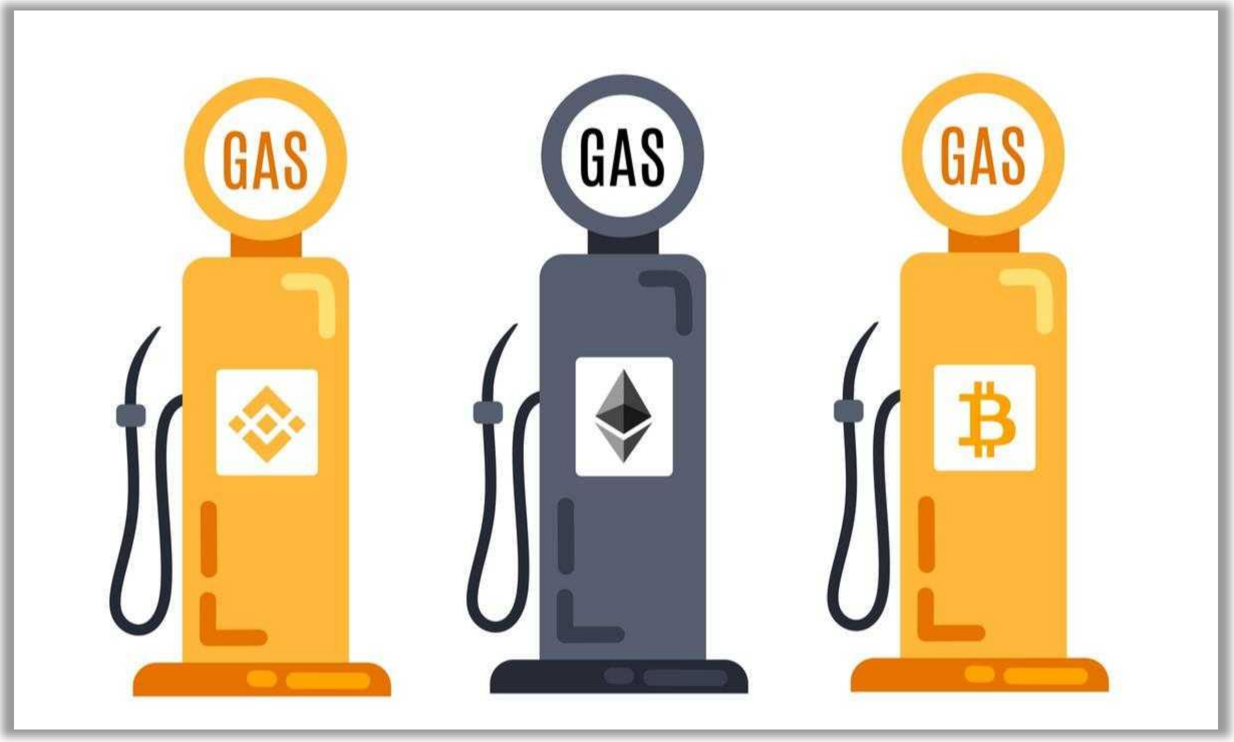
Solidity syntax is influenced by JavaScript, making it relatively easy for developers familiar with JavaScript or similar C-style languages to learn and work with Solidity.

Solidity is statically typed, meaning variable types are explicitly declared at compile time. This helps catch type-related errors and enhances the security and reliability of smart contracts.

Integrated Development Environment (IDE) Support: Solidity is supported by various development tools and IDEs, such as Remix, Truffle, and Visual Studio Code extensions.

These tools provide features like syntax highlighting, code completion, debugging, and testing to aid developers in writing, deploying, and testing Solidity-based smart contracts.

What is Gas in Ethereum Blockchain?



Gas is used to measure the complexity and cost of executing operations within the Ethereum Virtual Machine (EVM). Each operation, such as arithmetic calculations, data storage, or contract interactions, consumes a certain amount of gas.

Gas Price is another important factor that determines the cost of executing a transaction. It represents the amount of Ether (ETH) a user is willing to pay per unit of gas.

The purpose of using gas in Ethereum is to ensure that the network remains secure, prevents infinite loops, and discourages resource-intensive or malicious operations. It incentivizes users to optimize their code and use resources efficiently.

Gas has multiple associated terms with it:

- Gas Prices
- Gas Cost
- Gas Limit
- Gas Fees

The Principle behind Gas is to have a stable value for how much a transaction or consumption costs on the Ethereum Network.

Gas Price:

Gas price refers to the amount of Ether (ETH) a user is willing to pay per unit of gas when executing a transaction or invoking a smart contract on the Ethereum blockchain. It is denominated in Gwei, which is a subunit of Ether.

Gas Cost:

Gas cost refers to the actual amount of gas used during the execution of a transaction or smart contract. Each operation within the Ethereum Virtual Machine (EVM) consumes a specific amount of gas.

The gas cost is calculated by multiplying the gas used by the gas price.

Gas Limit:

Gas limit is the maximum amount of gas a user is willing to consume for a transaction or smart contract execution on the Ethereum network.

It represents the maximum computational steps that can be executed.

Gas Fees:

Gas fees are the charges incurred by users for utilizing computational resources on the Ethereum network.

It is the total cost in Ether (ETH) paid by a user to cover the gas used during transaction or smart contract execution.

What is Decentralized Autonomous Organization?

A Decentralized Autonomous Organization (DAO) is an organization or entity that operates on a decentralized blockchain network, such as Ethereum, and is governed by smart contracts and voting mechanisms rather than a centralized authority.

A DAO operates on a decentralized blockchain network, meaning it is not controlled or governed by a single central entity.

Since a DAO's operations are recorded on a blockchain, its activities are transparent and auditable. The transactions, proposals, and voting results are publicly visible and verifiable.

DAOs often have their own native tokens or tokens representing ownership or participation rights within the organization. These tokens can be used for voting, participating in decision-making, or earning rewards within the DAO ecosystem.

DAOs have a wide range of potential use cases. They can be utilized for decentralized investment funds, crowdfunding platforms, decentralized governance models for organizations or communities, decentralized marketplaces, and more.

Ethereum Use Cases:

Ethereum, as a versatile blockchain platform, has a wide range of use cases beyond its native cryptocurrency, Ether (ETH).

Here are some notable use cases of Ethereum:

- **Decentralized Finance (DeFi):** Ethereum has emerged as a leading platform for DeFi applications. DeFi encompasses various financial services, including lending, borrowing, decentralized exchanges, stablecoins, yield farming, and more.

- **Tokenization and Crowdfunding:** Ethereum allows the creation and management of tokens, facilitating crowdfunding campaigns through Initial Coin Offerings (ICOs) or Security Token Offerings (STOs).

- **Decentralized Applications (dApps):** Ethereum provides a platform for developers to build and deploy decentralized applications. These dApps can range from games and social media platforms to decentralized marketplaces and prediction markets.
- **Supply Chain and Logistics:** Ethereum's transparency and immutability make it suitable for supply chain management. By recording the movement and provenance of goods on the blockchain, Ethereum can enhance traceability, reduce fraud, and improve efficiency in supply chain processes.
- **Identity and Authentication:** Ethereum can be utilized for decentralized identity systems, providing users with more control over their personal information and eliminating the need for centralized authorities.

- **Gaming and Non-Fungible Tokens (NFTs):** Ethereum has become a popular platform for blockchain-based gaming and the creation of Non-Fungible Tokens (NFTs). NFTs are unique digital assets that can represent ownership of digital art, collectibles, virtual real estate, and more.

- **Governance and Voting:** Ethereum's smart contracts can facilitate decentralized governance models for organizations and communities. By using tokens and voting mechanisms, Ethereum-based systems allow participants to propose and vote on decisions, making the governance process more transparent and inclusive.

Nodes in Ethereum Blockchain:

In the Ethereum blockchain, a node refers to any computer or device that participates in the network by maintaining a copy of the blockchain and validating transactions. Nodes play a crucial role in the decentralized nature of the Ethereum network.

Software that can act as an Ethereum node include **Parity** and **Go-Ethereum (geth)**.

This software mostly, if not always, also provides wallet functionality and some node software also allows other programs to indirectly interact with the Blockchain.

Types of Nodes:

- ➢ Full Node
- ➢ Light Node
- ➢ Archive Node

Full Nodes: Full nodes are the most comprehensive type of nodes in the Ethereum network.

They maintain a complete copy of the entire blockchain and validate all transactions and smart contracts independently.

full nodes perform tasks such as verifying transaction signatures, executing smart contracts, and storing the entire transaction history.

They provide the highest level of security and decentralization as they independently validate the network state.

Light Nodes: Light nodes, also known as "light clients," do not store the entire blockchain but instead rely on full nodes for verification.

They maintain a partial copy of the blockchain, storing only the necessary data such as account balances and headers.

Light nodes are lightweight and consume fewer resources compared to full nodes, making them suitable for devices with limited storage or processing capabilities.

Mining Nodes: Mining nodes, or miners, are specialized nodes responsible for adding new blocks to the Ethereum blockchain.

Miners compete to solve computationally intensive puzzles called Proof of Work (PoW) algorithms to validate transactions, create new blocks, and earn block rewards.

Mining nodes require significant computational power and specialized hardware to perform mining operations efficiently.

Accounts in Ethereum Blockchain

Ethereum is a decentralized Blockchain with smart contracts functionality.

It has the largest market capitalization after Bitcoin and the most active Blockchain with a very good and active developer community.

Ether(ETH) is the native currency in Ethereum Blockchain.

In Ethereum, the state is made up of objects called accounts.

An account is identified by a 20-byte address.

There will be state transitions happening through accounts which is nothing but the direct transfer of information and value between accounts.

An Ethereum Account:

An Ethereum account consists of four fields:

Nonce:

This field is a counter to make sure transactions are processed only once on the Blockchain.

This indicates the number of transactions sent from an account.

In contract accounts, this field represents the number of contract created by the account.

Balance:

This field indicates accounts ether balance in wei.

(1 ETH = 1,000,000,000,000,000,000 wei (1018))

CodeHash:

For external accounts, this field is an empty string.

For smart contracts, it is the account's contract code.

StorageRoot:

Also Known as storage hash, it represents an account's storage and it is empty by default.

Ethereum has 2 types of accounts:

- Externally Owned Account
- Contract Accounts

Externally Owned Account:

These types of accounts are controlled by private keys.

These are user accounts.

For example, the account we create with Metamask.

Metamask calls eth_requestAccounts api on Ethereum Blockchain which returns a hexadecimal address.

You get a public address(your wallet address) by taking the last **20 bytes** of the hash of the hexadecimal address returned and adding ox to the beginning.

Contract Accounts:

The contract accounts are created when the contract is deployed on the Blockchain and unlike externally owned accounts, these accounts are controlled by code.

Differences between two types of Accounts:

- Both types of accounts can receive, send, hold ETH and ERC-20 Tokens (Token standard on Ethereum Blockchain) and interact with other smart contracts deployed on the Blockchain.

- The creation of externally owned accounts does not cost anything whereas contract accounts cost ETH, as they require storage on the Blockchain.

- External accounts can initiate transactions whereas the contract accounts can only send the transaction when it receives a transaction. When a contract account receives a transaction, the code triggers and can do different types of actions such as transfer ERC-20 token or ETH. create new smart contracts, or call other smart contracts.

- Transfer between two Externally owned accounts are limited to ETH transfers or ERC-20 Tokens.

What are Decentralized Applications?

DApps, short for Decentralized Applications, are software applications that run on a decentralized network, typically utilizing blockchain technology.

Unlike traditional applications that are built on centralized servers controlled by a single entity, DApps operate on a peer-to-peer network of computers or nodes, where data and processing power are distributed across multiple participants.

Key characteristics of DApps include:

- **Decentralization:** DApps are designed to operate without a central authority, making them resistant to censorship and single points of failure.

- **Open Source:** DApps typically have their source code available to the public, allowing anyone to review, verify, and contribute to the codebase.

- **Transparency:** DApps utilize transparent and immutable blockchain technology, enabling users to track and verify transactions and data on the network.

- **Tokenization:** Many DApps have their native tokens or cryptocurrencies that facilitate various functions within the application, such as utility, governance, or as a medium of exchange.

How does a DApp Work ?

DApps operate through a combination of smart contracts, decentralized networks, and user interfaces.

Here's a general overview of how a typical DApp works:

- **Smart Contracts:** DApps often utilize smart contracts, which are self-executing contracts with the terms of the agreement directly written into code.
  Smart contracts run on a blockchain network and automatically enforce the rules and conditions defined within them.

- **Decentralized Network:** DApps run on decentralized networks, usually based on blockchain technology. These networks consist of multiple nodes that store a copy of the blockchain's data and participate in the validation and consensus process.

- **User Interfaces:** DApps provide user interfaces (UIs) that allow users to interact with the application. The UI can be a web application, a mobile app, or a specialized software interface.

- **Data Storage:** DApps can store data in various ways. Some DApps utilize the blockchain itself to store data in a transparent and immutable manner.

- **Transactions and Consensus:** When a user interacts with a DApp, such as making a transaction or updating data, the action is submitted to the decentralized network. The transaction is processed and validated by the network's nodes through a consensus mechanism (e.g., proof-of-work or proof-of-stake). Once consensus is reached, the transaction is recorded on the blockchain, ensuring transparency and immutability.

Benefits of Decentralized Applications:

- Transparency: DApps operate on transparent and immutable blockchain networks. Every transaction and operation is recorded on the blockchain, providing a high level of transparency.

- Security: DApps leverage the security features of blockchain technology. The decentralized nature of the network makes DApps more resilient to hacking attempts and single points of failure.

  Data stored on the blockchain is encrypted and distributed across multiple nodes, making it difficult for attackers to compromise the system.

- Trust and Autonomy: DApps eliminate the need for intermediaries, such as centralized authorities or middlemen. Transactions and interactions within DApps are governed by smart contracts, which are executed automatically and transparently.

- Censorship Resistance: DApps are designed to be resistant to censorship and control. Since they operate on decentralized networks, there is no central authority that can dictate or restrict access to the application.

- User Ownership of Data: In many DApps, users have ownership and control over their data. Instead of handing over personal information to centralized platforms, users can store their data locally or on decentralized storage systems while still being able to interact with the DApp.

Drawbacks of Decentralized Applications:

- **Scalability:** DApps face scalability issues due to the inherent design of decentralized networks. As every node in the network must process and store the entire blockchain, the transaction processing capacity can be limited.
- **Development Complexity:** Building DApps requires specialized knowledge of blockchain technologies, smart contract programming, and decentralized protocols.
  Developing secure and robust DApps can be more complex and time-consuming than traditional application development.

- **Security Risks:** While blockchain technology provides inherent security features, DApps are not immune to security risks.
  Smart contracts can have vulnerabilities, and if exploited, can lead to financial losses or other adverse consequences. Additionally, the decentralized nature of DApps means that users are responsible for their private keys and securing their assets, which can be prone to human error or hacking attempts.
- **Energy Consumption:** Some DApps, especially those built on proof-of-work blockchains, require significant computational power and energy consumption for mining and transaction validation. This has raised concerns about the environmental impact of DApps and the sustainability of their operations.

What are example of DApps?

- **Decentralized Finance (DeFi) DApps:** DeFi has been one of the most active sectors in the DApp space. DApps like Compound, Aave, and Uniswap provide decentralized lending, borrowing, and trading services, allowing users to interact with financial protocols directly, without intermediaries.

- **Gaming DApps:** DApps are increasingly being used in the gaming industry to provide unique gaming experiences and enable true ownership of in-game assets.

  For example, **CryptoKitties** is a DApp where users can collect, breed, and trade virtual cats using blockchain technology.

- **Social Media DApps:** Decentralized social media platforms aim to provide users with control over their data and content. Steemit is a DApp where users can create and curate content and get rewarded with cryptocurrency.

- **Supply Chain DApps:** DApps can be used to track and verify supply chain information, ensuring transparency and authenticity.

  For example, **VeChain** is a DApp that focuses on supply chain management, enabling businesses to track and authenticate products throughout the supply chain using blockchain technology.

- **Decentralized Exchanges (DEX):** DEXs, such as Kyber Network and SushiSwap, are DApps that facilitate peer-to-peer trading of cryptocurrencies without the need for intermediaries. They provide users with control over their funds and enable seamless trading directly from personal wallets.

Initial Coin Offering (ICO)

An Initial Coin Offering (ICO) is a crowdfunding method used by cryptocurrency projects to raise funds for their development and operations.

It involves the issuance and sale of a project's native cryptocurrency tokens to investors and supporters in exchange for established cryptocurrencies like Bitcoin (BTC) or Ethereum (ETH) or sometimes even fiat currencies.

The project team creates a fixed supply of tokens based on a predetermined set of rules. These tokens serve various functions within the project's ecosystem, such as utility tokens for accessing services or governance tokens for voting on project decisions.

How an Initial Coin Offering (ICO) Works?

When a Cryptocurrency project wants to raise money through ICO, the project organizers first step is determining how they will structure the coin.

ICOs can be structured in a few different ways, including:

- **Project announcement:** The project team introduces their concept and goals, often through a whitepaper explaining the project's technical and business aspects.

- **Token creation:** The project team generates a fixed supply of tokens with specific rules and functionality within the project's ecosystem.

- **Marketing and promotion:** The project team promotes the ICO to attract potential investors, utilizing various channels such as social media, forums, and events.

- **Token sale:** Investors purchase the project's tokens using established cryptocurrencies or fiat currencies, typically through the project's website or a dedicated platform.

- **Fund allocation:** The funds raised during the ICO are utilized for project development, marketing, and operational expenses, as outlined in the project's whitepaper.

- **Token listing:** The project team seeks to list their tokens on cryptocurrency exchanges, enabling token holders to trade them with other cryptocurrencies.

Who can launch an ICO?

Anyone can launch an ICO. With very little regulation of ICOS in the U.S. Currently, anyone who can access the proper tech is free to launch a new Cryptocurrency.

But this lack of regulation also means that someone might do whatever it takes to make you believe they have a legitimate ICO and can abscond with the money.

Of all the possible funding avenues, an ICO is probably one of the easiest way to set up as a scam.

Initial Coin Offering vs Initial Public Offering

The primary difference between an ICO and an Initial Public Offering is that investing in an ICO does not secure an ownership stake in the Crypto project or company.

ICO participants are gambling that a currently worthless. currency will later increase in value above its original purchase price.

Though IPOs are funded by generally more conservative investors anticipating a financial return, ICOs may receive funding from risk-tolerant supporters keen to invest in a new, existing project.

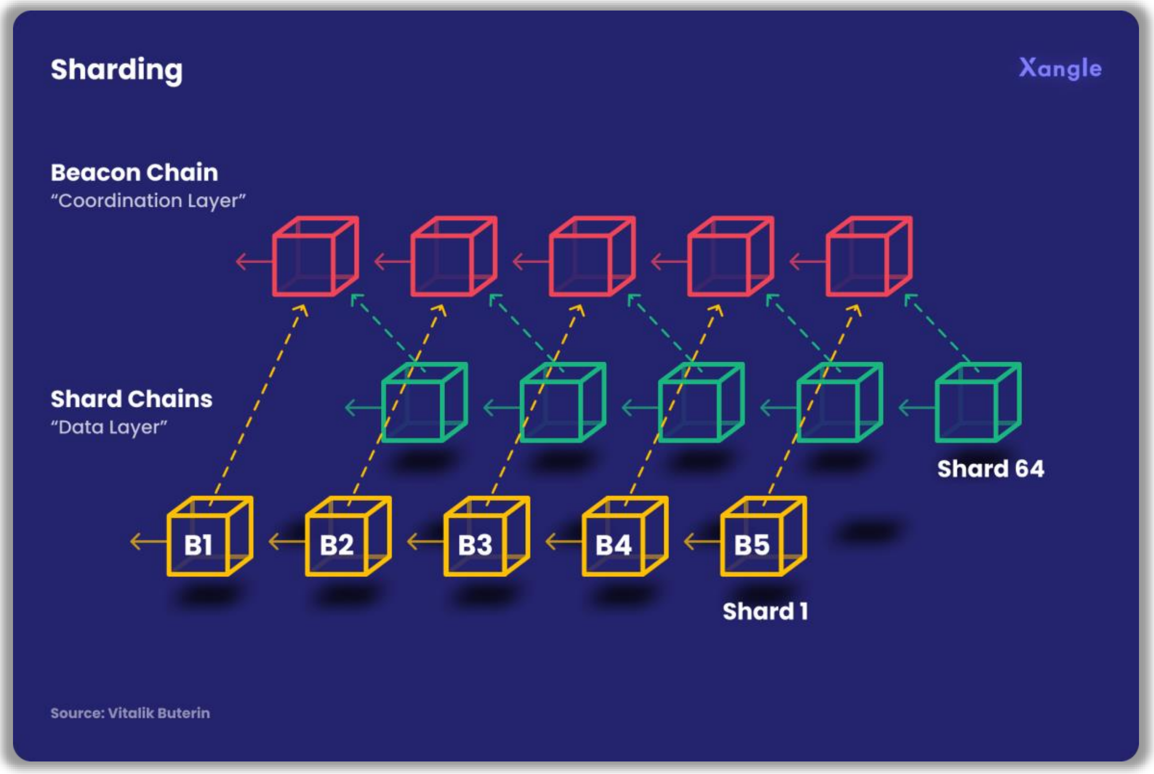
An ICO differs from a Crowdfunding event because it offers the possibility of financial gain over time, whereas crowdfunding initiatives receive donations. ICOs are also referred to as crowdsales.

What is Sharding?

Sharding is a technique used in blockchain technology to improve scalability and increase transaction processing capacity.

It involves partitioning the **blockchain network** into smaller subsets called "shards," each capable of processing its own set of transactions.

The blockchain network is divided into multiple shards, with each shard having its own set of nodes responsible for maintaining the shard's state and validating transactions within it.



The benefits of sharding include increased transaction throughput, reduced transaction confirmation times, and improved overall scalability of the blockchain network.

Sharding in the Blockchain is aimed to spread the load over the decentralized network by partitioning data through shards.

How does Sharding work?

- **Partitioning:** The blockchain network is divided into smaller subsets called shards, each responsible for processing a portion of the network's transactions.

- **Shard Independence:** Each shard operates independently, with its own set of nodes and transaction history. This allows for parallel processing of transactions, increasing the overall throughput of the blockchain.

- **Transaction Assignment:** When a transaction is initiated, it is assigned to a specific shard based on predefined criteria, such as the account involved or transaction type.

- **Shard Validation:** The assigned shard validates the transaction within its own network, ensuring correctness and consistency within its subset of the blockchain.

- **Inter-Shard Communication:** Shards need to communicate and share relevant information to maintain the integrity of the entire blockchain. Inter-shard communication protocols allow for data exchange and consensus across shards.

- **Consensus Mechanisms:** Sharding introduces new challenges to achieving consensus. Consensus mechanisms specific to sharded blockchains, such as cross-shard validation, are employed to ensure agreement on the state of the overall blockchain.

- **Security Considerations:** Sharding introduces potential security risks, such as attacks targeting specific shards. Security measures, like secure cross-links, are implemented to maintain the overall security of the blockchain network.

Advantages of Sharding:

- Using Sharding, we don't need and powerful and expensive computers.
- With Sharding, transactions per second will increase.
- With the help of Sharding, more number of Validators can join.
- Implementing Sharding will reduce huge energy consumption.

Disadvantages of Sharding:

But Sharding does not solve all the scaling and performance issues.

One big disadvantage is that the communication between shards is quite difficult to achieve.

What is Ethereum Gas?

Ethereum gas refers to the pricing mechanism and unit of measurement for computational work performed on the Ethereum blockchain.

- **Gas as a Unit:** Gas is the unit used to measure the computational work required to execute transactions and smart contracts on the Ethereum network.

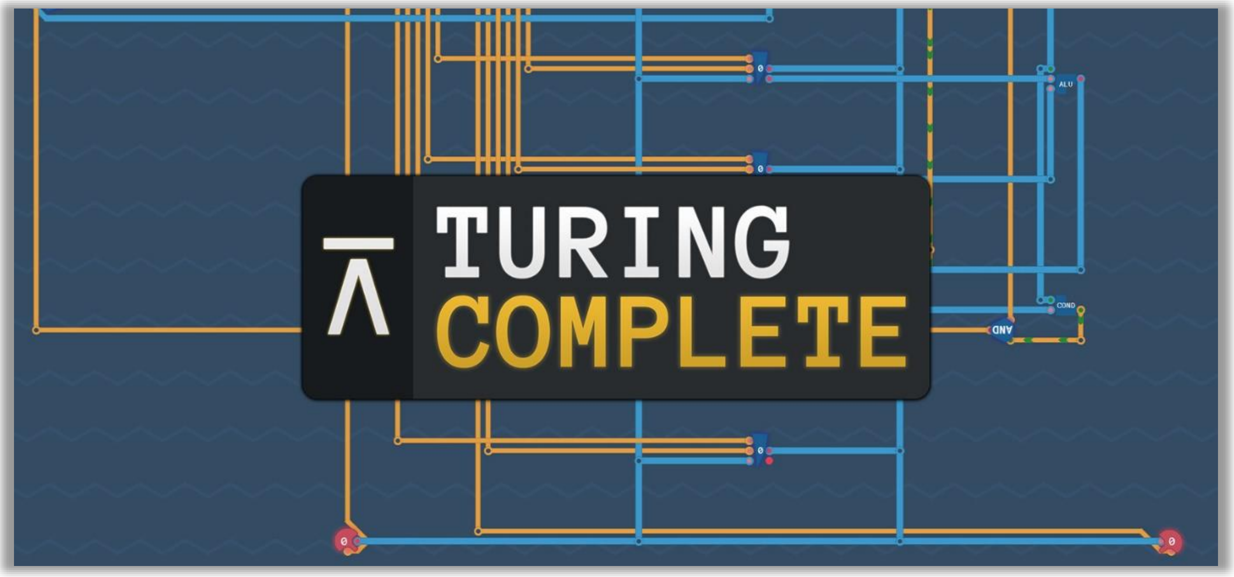
- **Gas Costs:** Each operation in an Ethereum transaction or smart contract execution consumes a certain amount of gas, which represents the computational resources utilized.

- **Gas Price:** Gas has an associated price denominated in ether (ETH). Gas price determines the cost in ETH that users are willing to pay for each unit of gas consumed.

- **Gas Limit:** The gas limit is the maximum amount of gas that a user is willing to spend on a transaction or smart contract execution. It represents the upper limit of computational work allowed.

- **Transaction Fees:** The total fee for an Ethereum transaction is calculated by multiplying the gas price by the gas used. This fee is paid by the sender to incentivize miners to include the transaction in a block and process it.

What is Turing Complete?



Turing completeness is a term used in computer science and mathematics to describe a system or programming language that can perform any computation that a Turing machine can, given enough time and resources.

- Turing Machine: A Turing machine is an abstract mathematical model that represents a hypothetical computing device.
  It consists of an infinite tape divided into cells and a head that can read and write symbols on the tape, as well as move left or right along the tape.

- Computation Power: Turing completeness refers to the ability of a system, language, or programming environment to simulate or replicate the behavior of a Turing machine. This means that any computation that can be done by a Turing machine can also be done by a system that is Turing complete.

- Loops and Conditionals: A Turing complete system must support the ability to perform loops (repetitive execution) and conditionals (branching based on logical conditions). These features allow for arbitrary computations and decision-making within the system.

- Universal Computation: Being Turing complete means that a system has the capability to solve a wide range of computational problems, including those that involve complex algorithms and computations.