Deploying Smart Contracts in Blockchain

Overview

Introduction

What is a Smart Contract?

- A smart contract is a self-executing contract with the terms of the agreement directly written lines of code.
- It automatically executes, controls, or documents legally relevant events and actions according to the terms of the contract.

Why Blockchain?

- Blockchain's **decentralized** and **secure** nature makes it ideal for deploying smart contracts.
- These contracts are **immutable** and transparent, ensuring trust between parties without the need for intermediaries.
- Blockchain enables the automation of processes through smart contracts, reducing the need for **manual intervention**.
- Once conditions are met, the contract automatically executes the agreed-upon actions, increasing efficiency and reducing delays or human errors.
- By removing intermediaries (such as banks or lawyers), blockchain-based smart contracts can significantly **lower transaction fees** and administrative costs. This makes it a more affordable alternative for conducting business and executing agreements.

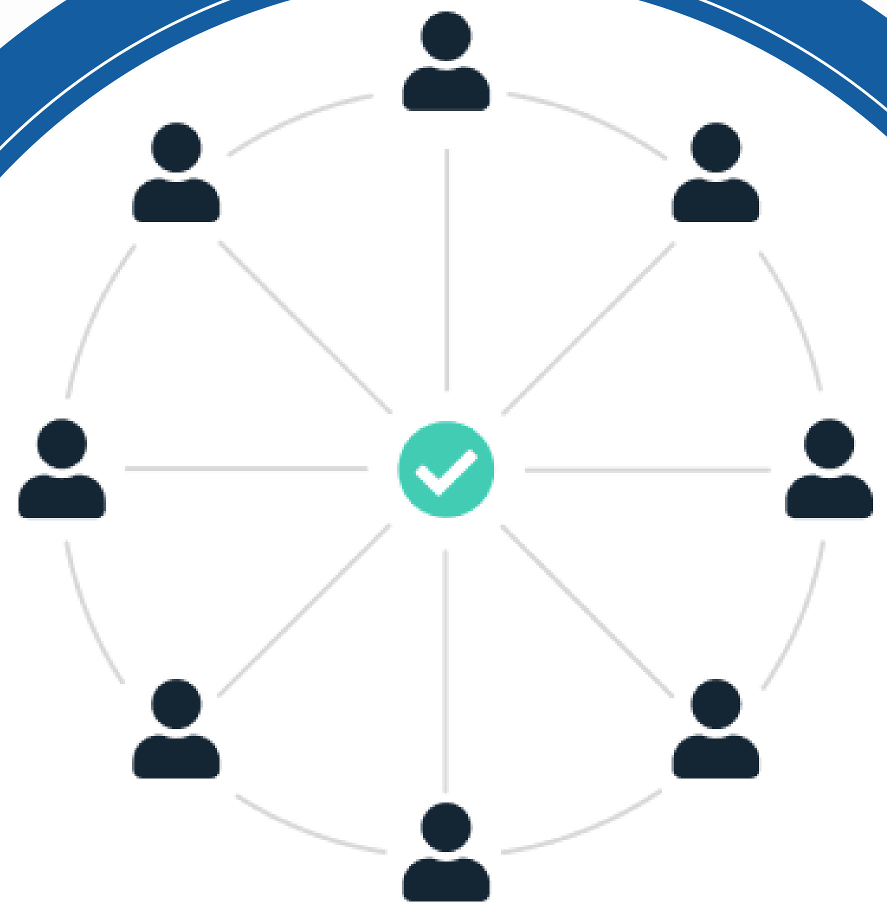
Blockchain

Decentralization

- Blockchain operates without a central authority, distributing control across a network of nodes.
- Participants interact directly in a **peer-to-peer** network, removing intermediaries and enabling trustless transactions.
- Trust in the system is maintained through cryptography and consensus mechanisms, not reliance on a **central** entity.

Consensus

- Consensus ensures that all nodes in the network agree on the validity of transactions and the current state of the blockchain.
- **Proof** of **Work** (PoW) involves miners solving complex mathematical problems to validate and add new blocks to the blockchain.
- **Proof** of **Stake** (PoS) selects validators to add blocks based on the amount of cryptocurrency.



Ethereum

- Ethereum is an open-source, decentralized blockchain platform designed for building and deploying smart contracts and decentralized applications (DApps).
- It provides a global computing environment where developers can create programmable, trustless applications.

Blockchain

enables peer-to-peer interactions without the need for intermediaries. It operates on a decentralized network, allowing secure and transparent execution of smart contracts. Unlike Bitcoin, Ethereum's primary purpose is to support decentralized applications and smart contracts.

Virtual Machine

The EVM is the environment that executes smart contracts and DApps on Ethereum. It ensures that smart contracts are processed consistently across all nodes in the network, maintaining the integrity of execution through its decentralized consensus.

Smart Contracts

Smart contracts on Ethereum are self-executing agreements where the terms are written directly into the code. They allow for automatic execution of transactions when predefined conditions are met, providing security, efficiency, and transparency without intermediaries.

- Ethereum uses its native cryptocurrency, Ether (ETH), to power transactions and contract execution within its ecosystem.

Concepts of Bitcoin

Decentralized Digital Currency

Bitcoin is a decentralized digital currency that operates on a peer-to-peer network, allowing users to send and receive transactions without the need for intermediaries like banks. It uses blockchain technology to record and verify transactions.

Bitcoin's Blockchain

The Bitcoin blockchain is a distributed ledger that records all transactions in a secure and transparent manner. Each transaction is grouped into blocks, which are then added to a chain, forming a permanent and immutable history of transactions.

Mining and Proof-of-Work (PoW)

Bitcoin relies on mining, where miners solve complex cryptographic puzzles to validate transactions and add them to the blockchain. This process, known as Proof-of-Work (PoW), ensures security and integrity within the network.

Platforms

- Blockchain platforms that support smart contracts provide decentralized environments where self-executing contracts can be deployed and run autonomously.

Ethereum

It provides the Ethereum Virtual Machine (EVM), which processes contract execution across the network. Developers use Solidity to write smart contracts, which are then deployed and run autonomously on the blockchain.

Hyperledger

It allows smart contracts, called chain code, to be deployed in a permissioned blockchain environment. It supports multiple programming languages like Go, Java, and JavaScript and allows for fine-grained access control.

Stellar

Its smart contracts are lightweight and primarily handle payment-related functions. Stellar uses a simpler scripting language to enable quick and cost-effective execution of basic contract logic related to asset exchange and payment settlement.

Languages

Smart contracts are executed by blockchain platforms through various programming languages. These languages are designed to ensure determinism, security, and efficiency in the decentralized environment of a blockchain.

Solidity

Primary language for Ethereum smart contracts. It is a contract-oriented, high-level language resembling JavaScript and C++.

Features:

- Used for decentralized apps (DApps).
- Handles complex contract logic like token creation.

Chain code

Smart contracts in Hyperledger Fabric, written in Go, Java, or JavaScript, used for private enterprise blockchains.

Features:

- Focused on business processes.
- Works in permissioned blockchains.

Vyper

A Python-based alternative to Solidity, focused on security and simplicity for Ethereum smart contracts.

Features:

- Emphasizes readability and security.
- Fewer features to reduce complexity and risks.

Rust

Used for smart contracts on Solana. Known for high performance, memory safety, and concurrency.

Features:

- Ensures scalability and security.
- Ideal for high-throughput applications.

DAO (Decentralized Autonomous Organization)

Structure & Governance

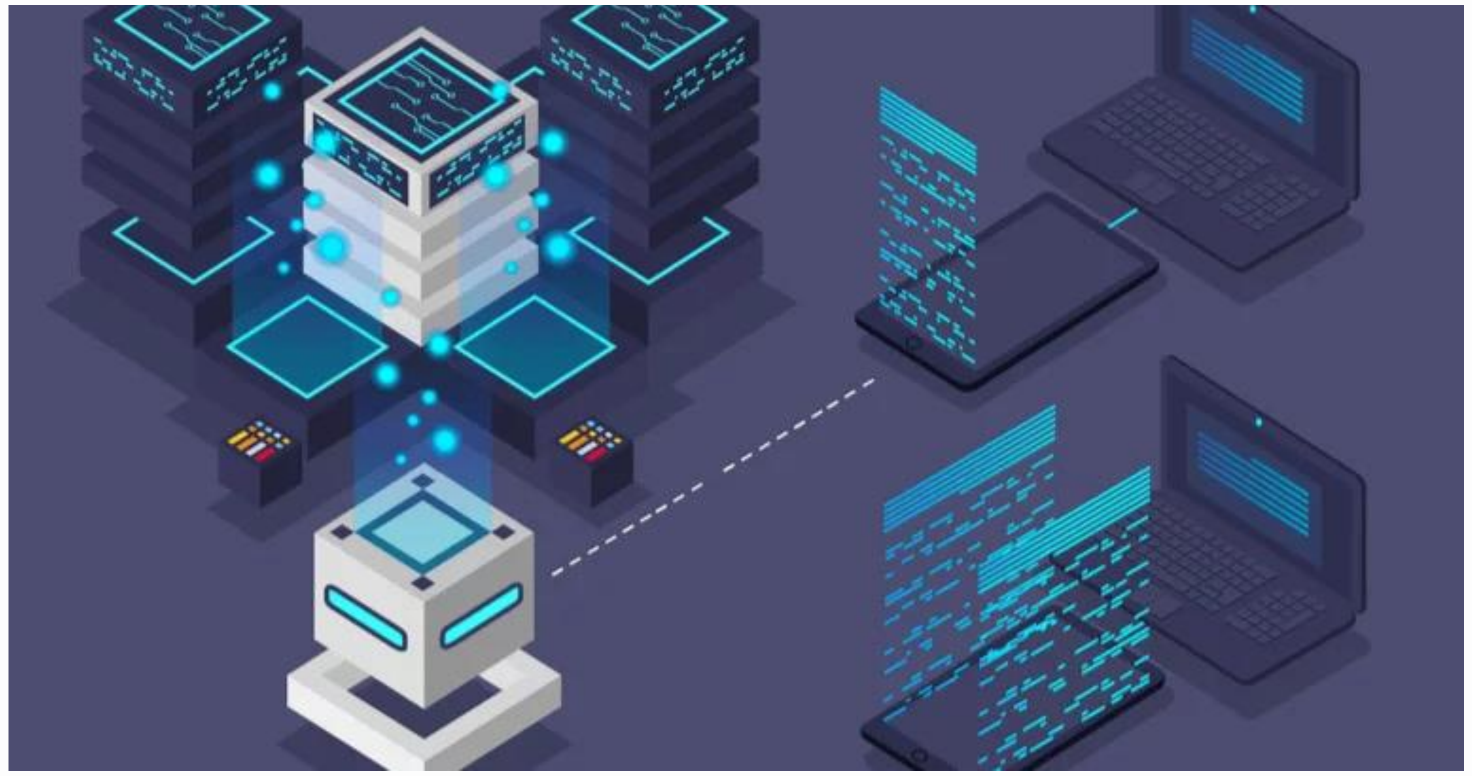
DAOs are designed to operate without a central authority. Decision-making is carried out through voting mechanisms, with members participating based on their token holdings or other governance models.

Token-based voting: Members vote on proposals based on the number of tokens they hold.
Transparent: All decisions and activities are visible on the blockchain, ensuring accountability.

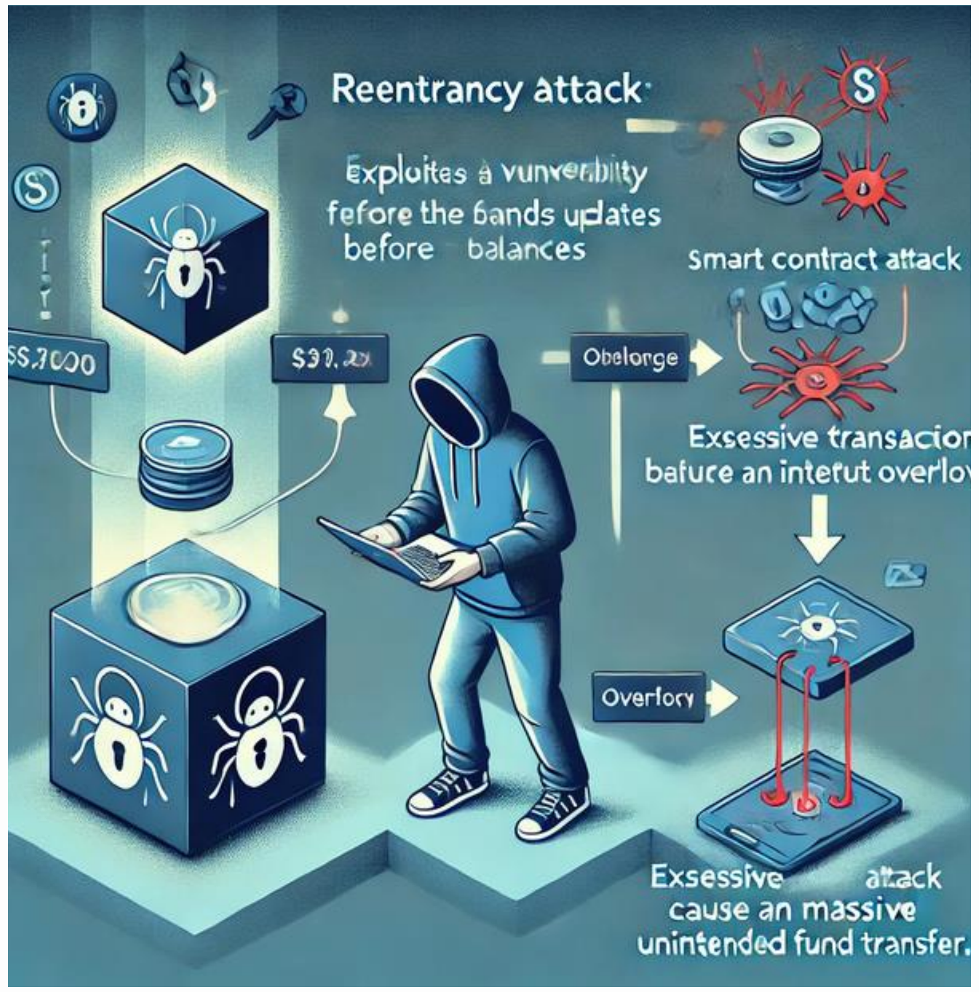
Purpose & Functionality

DAOs serve various functions, from managing investment funds to overseeing decentralized applications (DApps) or collective ventures. They operate autonomously, executing smart contract-coded rules and decisions.

Autonomous Execution: Smart contracts automatically execute decisions.
Open to All: Anyone can participate in the DAO's governance by acquiring tokens.



Security in Smart Contracts and DAOs


Reentrancy attack
Exploites a vunverbilty fefore the bands updates before balances
Smart contract attack
Obelonge
Exsessive transacion bafuce an intetut overloy
Overfory
Exsessive attack cause an massive unintended fund transfer.

Smart Contract Vulnerabilities

Smart contracts are only as secure as the code that powers them. Bugs or poorly written code can lead to security risks, such as reentrancy attacks, integer overflows, or logic flaws.

Risks

Reentrancy Attack: Occurs when a contract calls another contract before finishing its own execution, allowing attackers to withdraw funds multiple times.

DAO Security Concerns

DAOs rely on smart contracts to execute decisions autonomously. Security flaws in the DAO's code can be exploited to manipulate votes or steal funds, as seen in the 2016 DAO Hack.

Risks

Governance Attacks: Attackers may control governance votes or manipulate the decision-making process.

Mitigation Strategies

To prevent security breaches, developers and organizations must implement best practices in coding, testing, and monitoring.

Risks

Code Audits: Regular, in-depth reviews of code by third-party experts to identify vulnerabilities.

THANK YOU!