UNIT-2 NOTES
BLOCKCHAIN TECHNOLOGY
20 CST/ITT-412

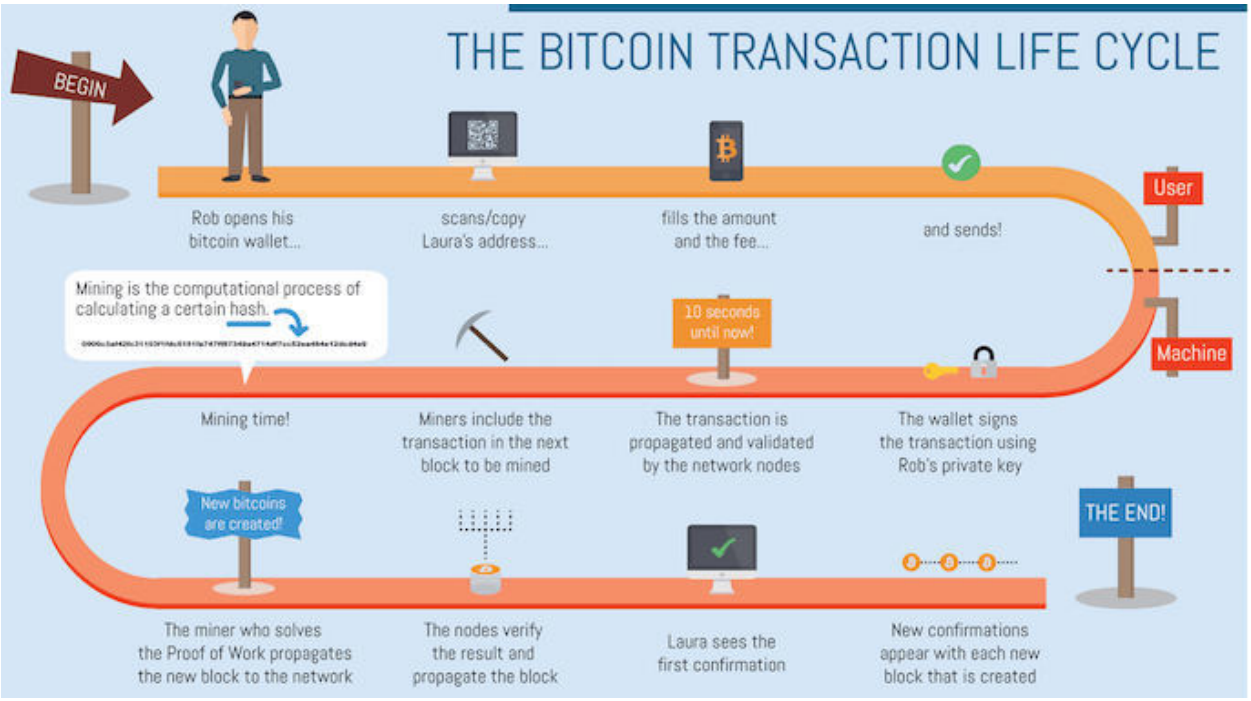
Prepared by:
Ankita Sharma
MSC Block chain technology
Course code - 20 CST/ITT-412

"DIAGRAMS ARE COMPULSORY"

Introduction to Bitcoin:

Bitcoin is a groundbreaking digital currency that has transformed the world of finance and technology. It was created by an anonymous entity or group using the pseudonym "Satoshi Nakamoto" in 2008, and the Bitcoin network was launched as open-source software in 2009. Bitcoin is often referred to as a cryptocurrency because it uses cryptographic techniques to secure transactions and control the creation of new units.

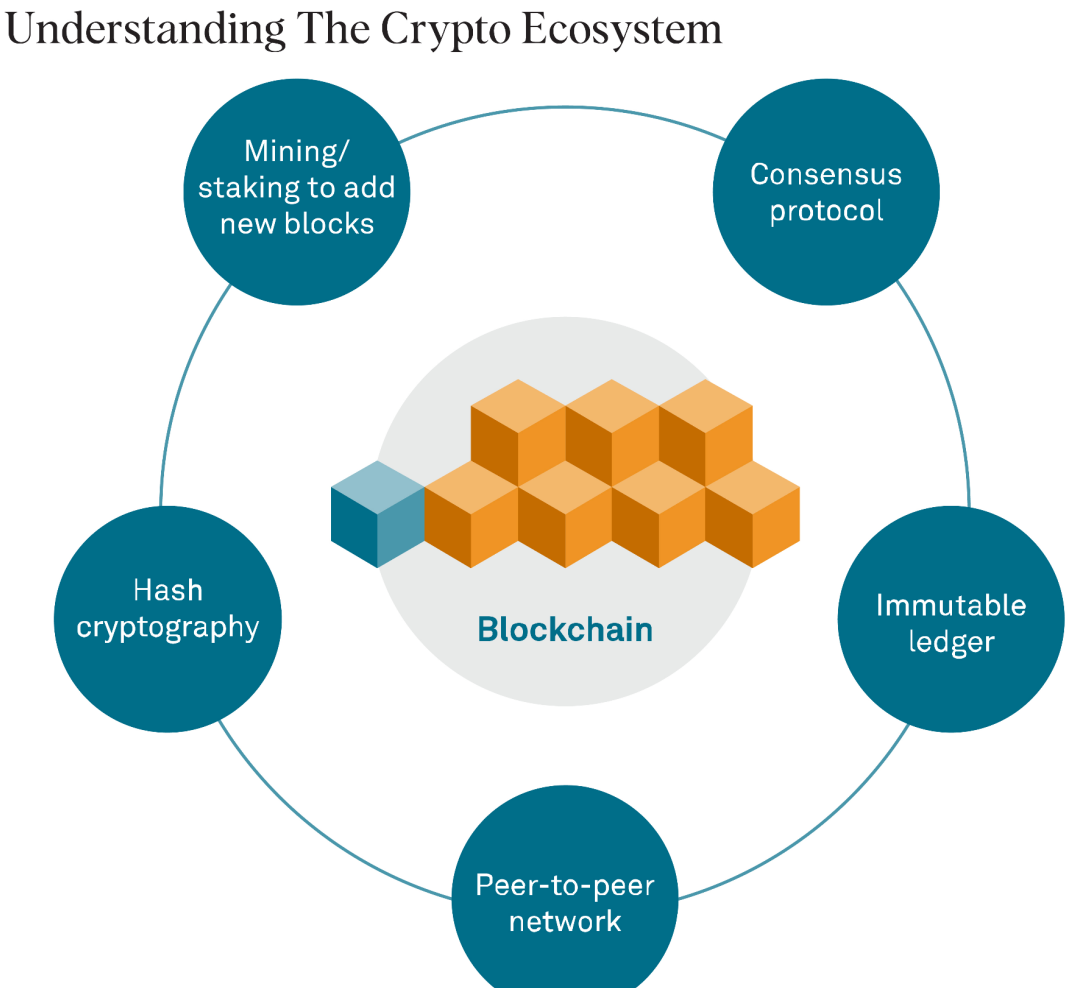
At its core, Bitcoin is a decentralized digital ledger technology known as a blockchain. A blockchain is a distributed and immutable ledger that records all transactions across a network of computers. This technology eliminates the need for a central authority, such as a bank or government, to validate and record transactions.



Origins of Bitcoin and the Whitepaper by Satoshi Nakamoto

The origins of Bitcoin remain shrouded in mystery. Satoshi Nakamoto's true identity is still unknown, and this anonymity was intentional. Nakamoto's motivation for creating Bitcoin was to address long-standing issues with the traditional financial system. The whitepaper titled "Bitcoin: A Peer-to-Peer Electronic Cash System," published by Nakamoto in October 2008, laid out the conceptual framework for Bitcoin.

The whitepaper proposed a solution to the "double-spending problem," which had hindered previous attempts to create digital currencies. It introduced the concept of a decentralized network of nodes (computers) that collectively maintained a public ledger of transactions, ensuring the security and integrity of the system.

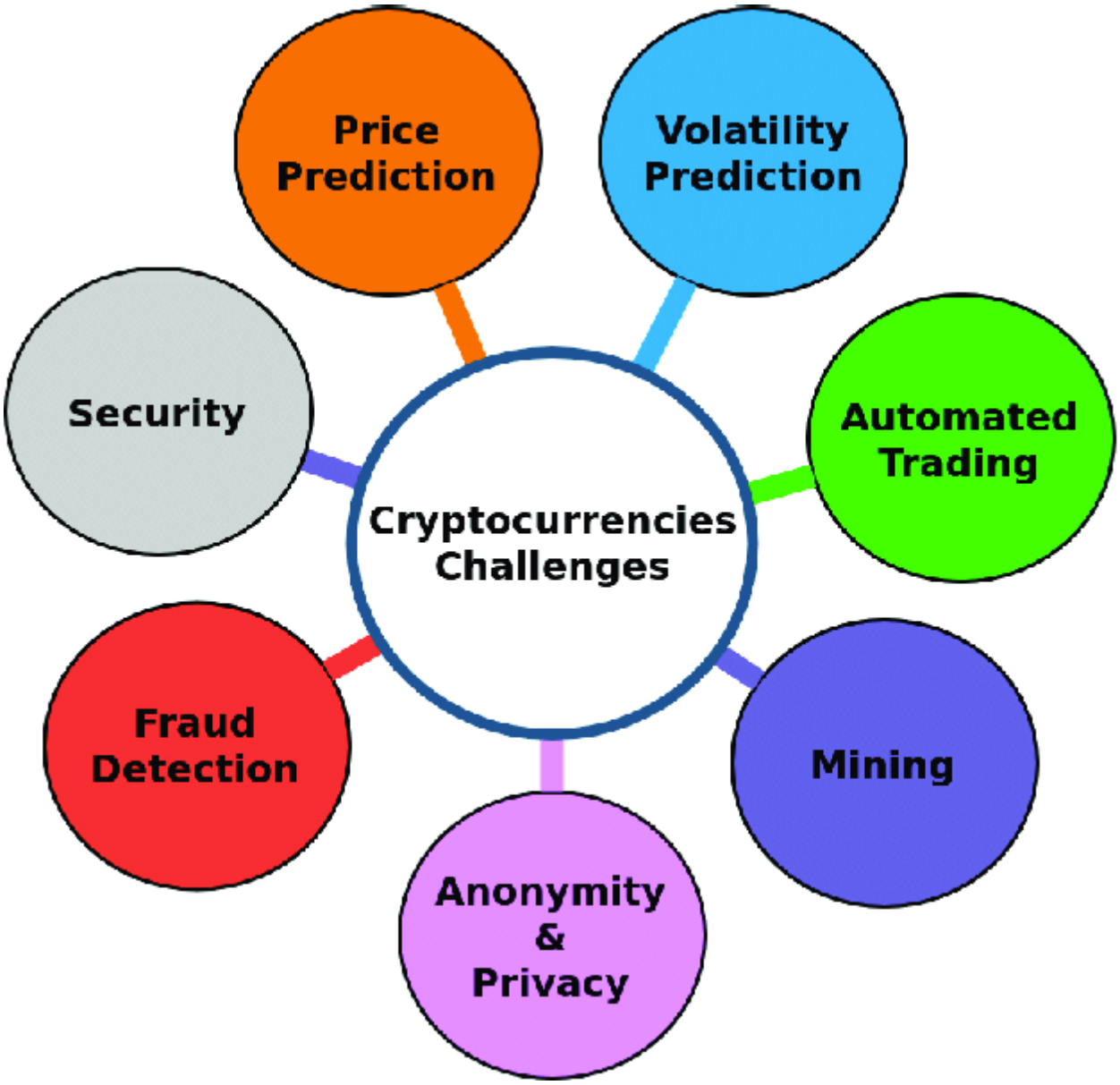


The key components of Nakamoto's whitepaper include the description of the blockchain, proof-of-work mining, cryptographic techniques for securing transactions, and the idea of a peer-to-peer electronic cash system. This paper served as the blueprint for the development of Bitcoin and inspired the creation of numerous other cryptocurrencies.

The Challenges

Bitcoin has faced several challenges and criticisms since its inception:

1. Scalability;- Bitcoin's original design limits the number of transactions it can process per second, leading to slow confirmation times and high transaction fees during periods of high demand.

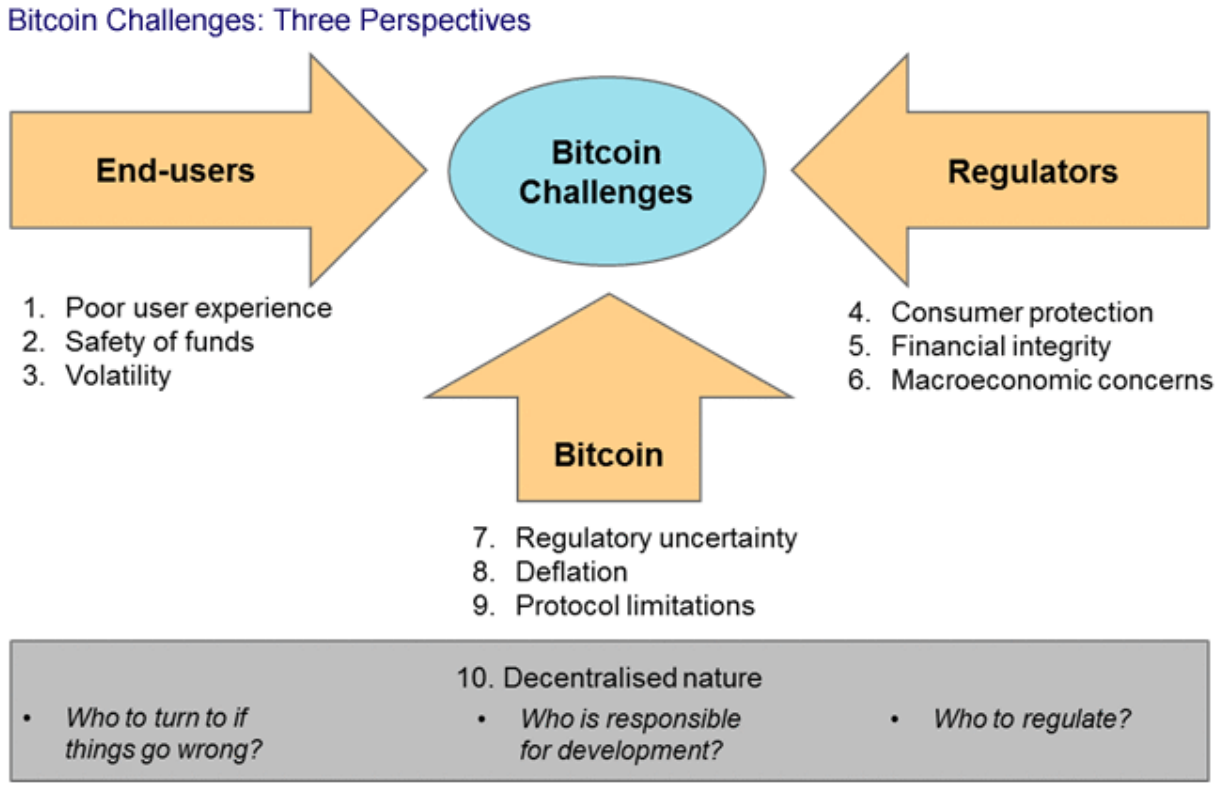


2. Regulatory Scrutiny:- Governments and regulatory bodies worldwide have expressed concerns about the potential misuse of Bitcoin for illegal activities, leading to regulatory actions and uncertainty.

3. Energy Consumption:- Bitcoin mining, based on the proof-of-work consensus mechanism, is energy-intensive. Critics argue that it has a significant environmental impact.

4. Security Concerns:- While Bitcoin's blockchain is highly secure, individual users can be vulnerable to hacks, scams, and phishing attacks if they do not take proper precautions.

5. Volatility:- Bitcoin's price can be extremely volatile, which makes it a risky asset for both investors and users as a medium of exchange.



Structure and Components of a Bitcoin Transaction

A Bitcoin transaction is a fundamental unit of activity on the Bitcoin network. It consists of several key components:

1. Input: An input references a previous transaction output and provides the funds for the current transaction. It includes the amount of Bitcoin being spent and a script that verifies the transaction's legitimacy.

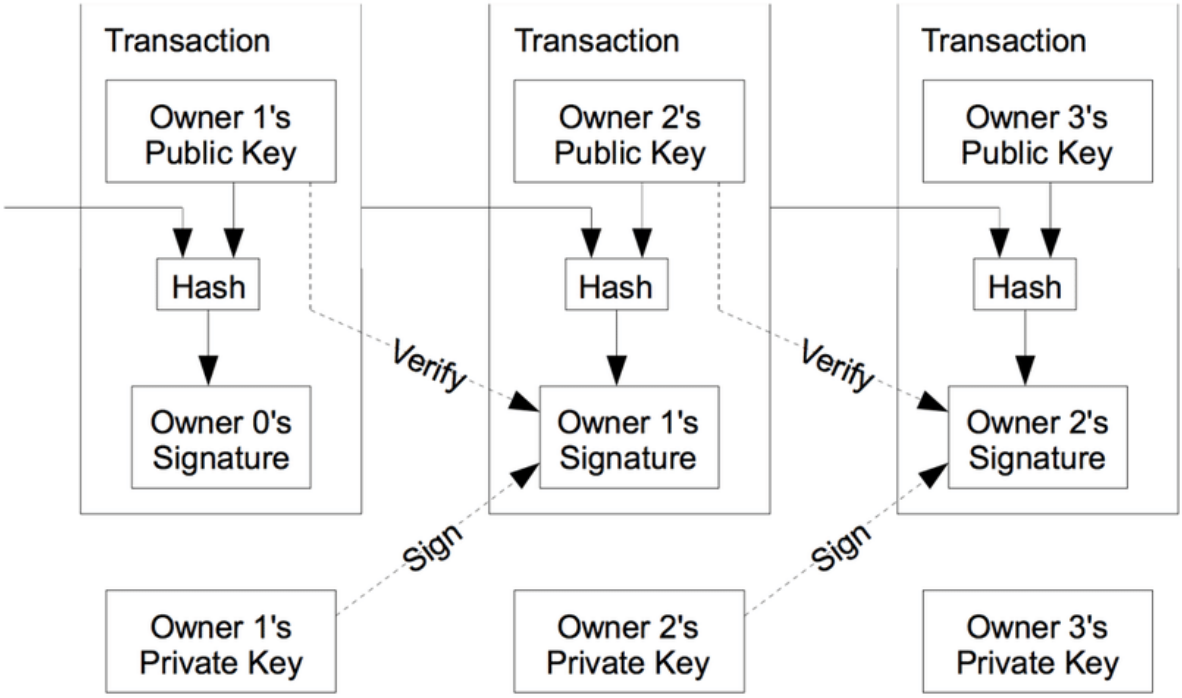
2. Output: An output specifies the recipient of the Bitcoin and the amount to be received. It also contains a locking script that defines the conditions under which the recipient can spend the received funds.

3. Transaction ID:- Each transaction has a unique identifier known as a Transaction ID (TxID). This helps in tracking and verifying transactions on the blockchain.

4. Digital Signatures:- Digital signatures are used to prove the ownership and authorization of the funds being spent. The sender signs the transaction with their private key, and the recipient can verify it using the sender's public key.

5. **Transaction Fee:-** To incentivize miners to include their transactions in the next block, users may attach a transaction fee. Higher fees typically result in faster confirmation times.

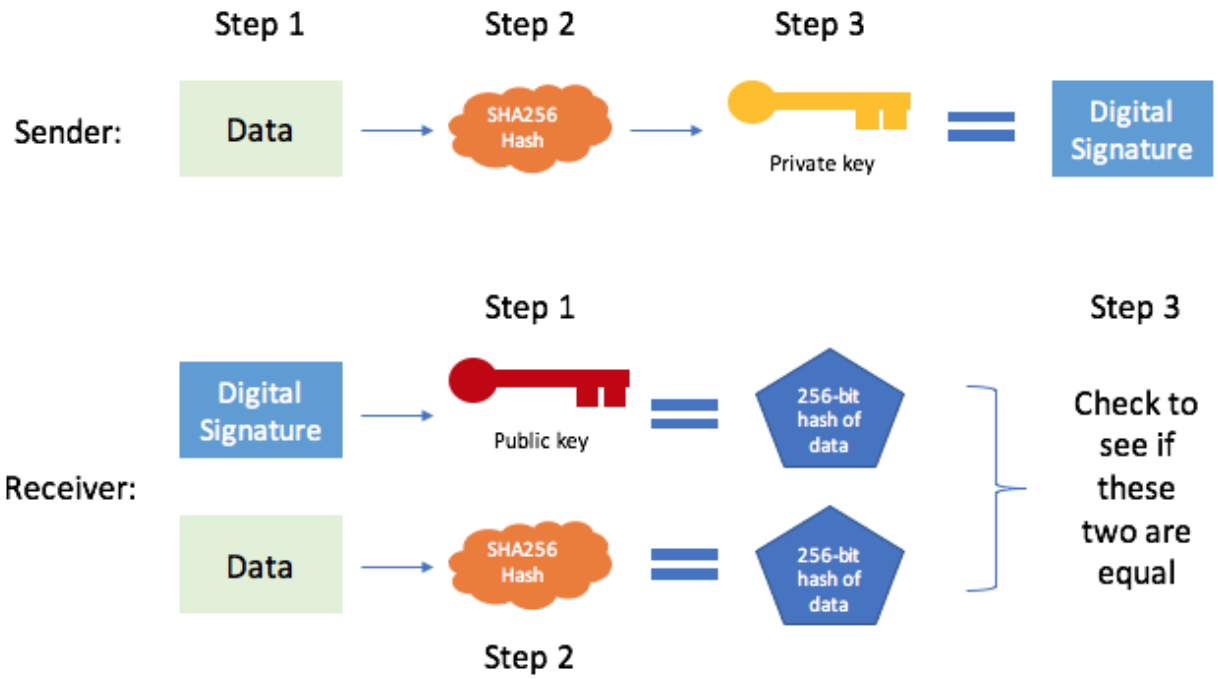
6. Change Output:- When a transaction spends less than the total amount of a previous input, the remaining funds are sent back to the sender as "change" in a new output.



Bitcoin Scripting Language and Their Uses

Bitcoin scripting language is a simple and stack-based programming language used to define the conditions under which a Bitcoin transaction can be spent. It enables the creation of various transaction types and conditions, including:

1. Pay-to-Public-Key-Hash (P2PKH):- This is the most common transaction type, where the recipient's public key hash is specified, and they can spend the funds by providing a valid digital signature.



2. Pay-to-Script-Hash (P2SH):- P2SH allows more complex conditions to be defined in a separate script. The recipient redeems the funds by providing a script that matches the hash specified in the transaction output.

3. Multisignature Wallets:- Bitcoin scripting allows multiple public keys to be involved in a transaction. This is often used for increased security, requiring multiple signatures to spend the funds.

4.Time-Locked Transactions:- Bitcoin scripting enables the creation of time-locked transactions, where funds can only be spent after a specified block height or timestamp.

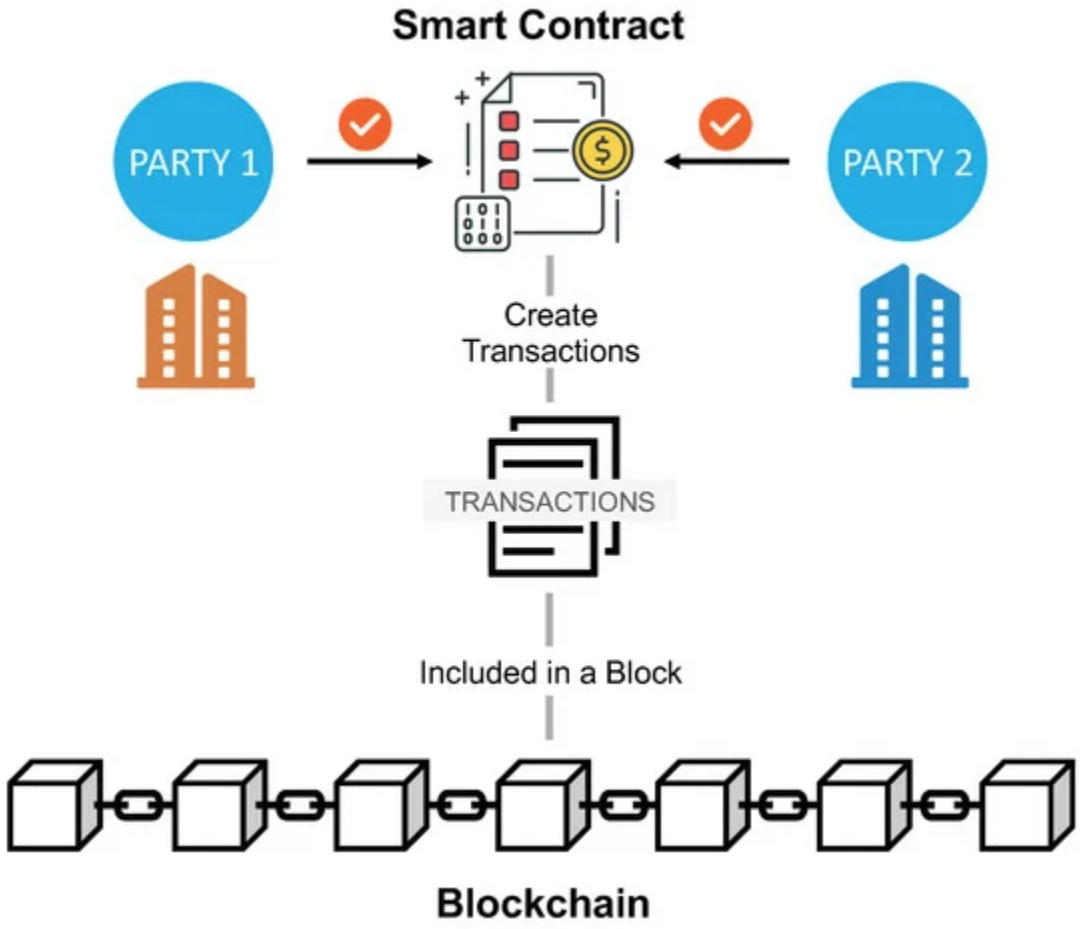
5. Escrow Services:- Bitcoin scripting can be used to create escrow services, where a third party (an arbiter) must approve the transaction for it to be valid.

6. Atomic Swaps:- With Bitcoin scripting, it's possible to perform atomic swaps, enabling the trustless exchange of Bitcoin for other cryptocurrencies without the need for an intermediary.

Blockchain 2.0: Ethereum and Smart Contracts" in detail:

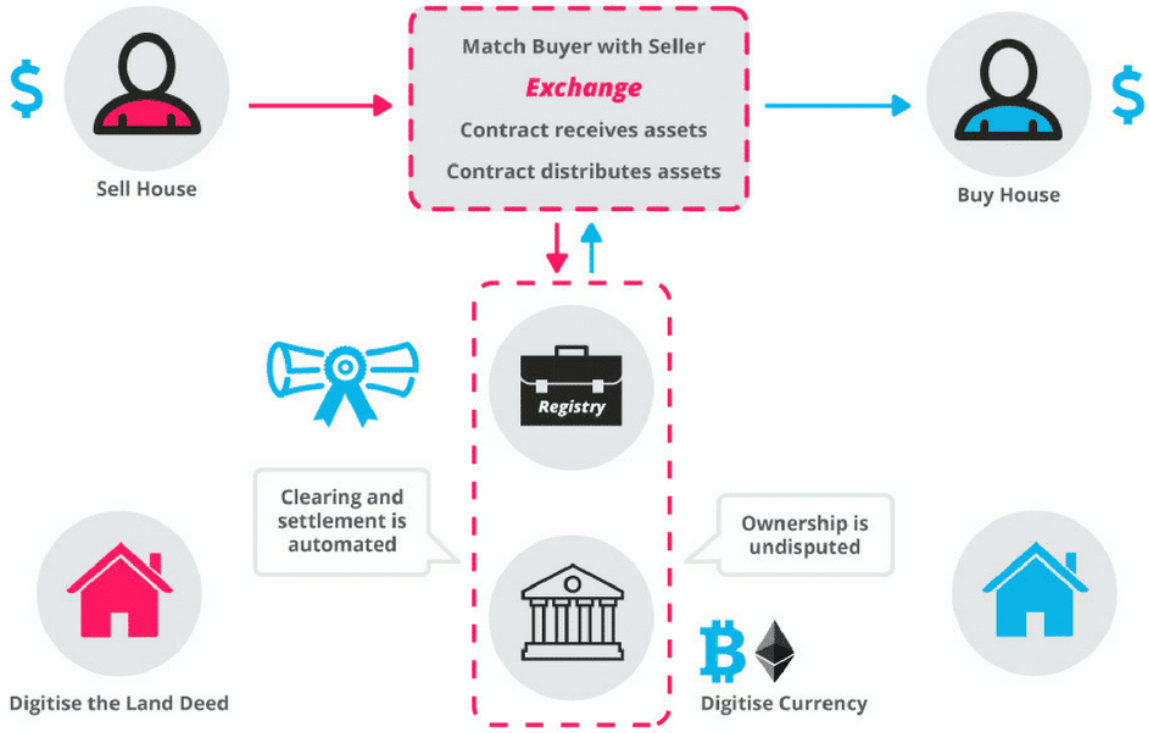
Ethereum and Smart Contracts

Ethereum, often referred to as Blockchain 2.0, represents a significant evolution beyond the capabilities of Bitcoin. While Bitcoin primarily serves as a digital currency, Ethereum extends blockchain technology to offer a broader range of functionalities through the concept of smart contracts.



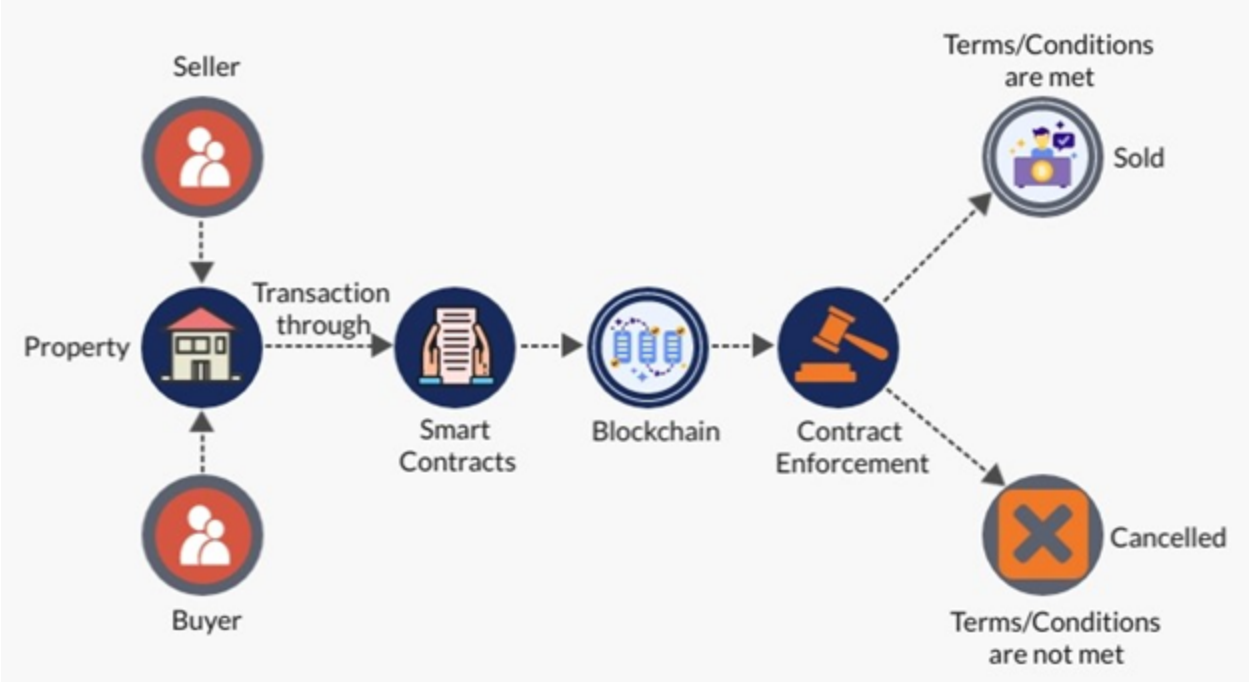
Smart Contracts:- Smart contracts are self-executing contracts with the terms and conditions of the agreement directly written into code. They are stored on the Ethereum blockchain and automatically execute when predefined conditions are

met. Smart contracts eliminate the need for intermediaries in various types of agreements, as they can automate and enforce contract execution.



Turing Completeness of Smart Contract Languages and Verification Challenges

One of the defining features of Ethereum's smart contract system is the Turing completeness of its programming languages, particularly Solidity. Turing completeness means that a programming language can perform any computation that a Turing machine can. In the context of smart contracts, this implies that Solidity can express a wide range of complex algorithms and operations.



Verification Challenges:- While the Turing completeness of Solidity provides developers with immense flexibility and power, it also introduces significant challenges in terms of verification and security. Writing secure smart contracts is crucial because once deployed on the Ethereum blockchain, they are immutable and cannot be changed. Some common verification challenges include:

1.Security Vulnerabilities:- Smart contracts can be vulnerable to various security issues, including reentrancy attacks, integer overflows, and unauthorized access to contract functions. Thorough testing and auditing are essential to identify and mitigate these vulnerabilities.

2.Complexity:- Complex smart contracts may have intricate interdependencies and logic, making it difficult to ensure their correctness and security. Tools and techniques for formal verification are necessary to verify the behavior of such contracts.

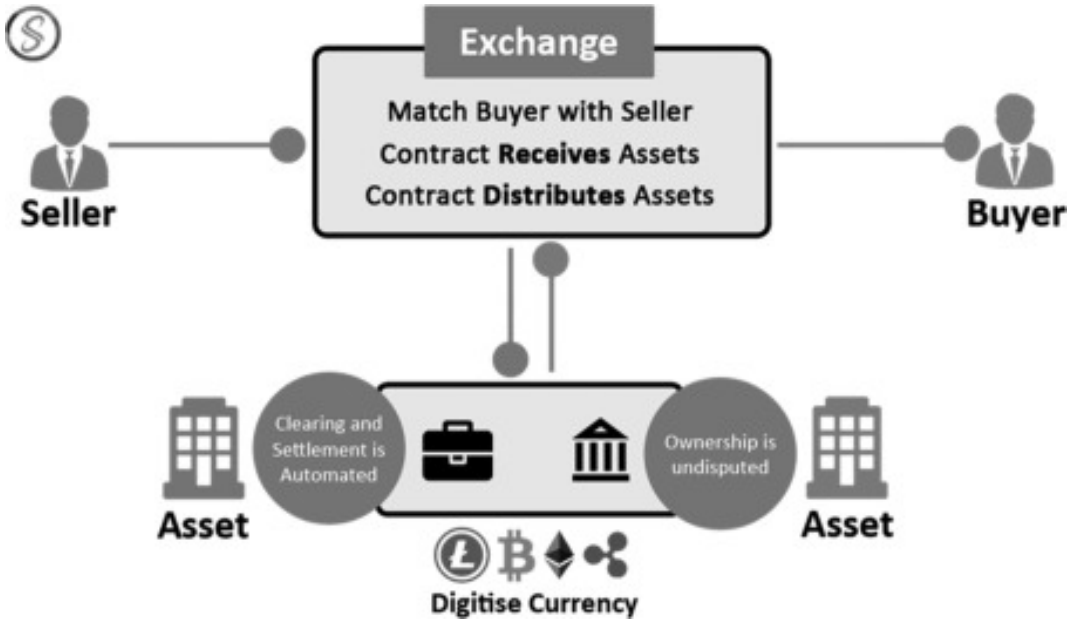
3.Upgradability: Making upgrades to deployed smart contracts without introducing vulnerabilities or disrupting the contract's existing functionality is a

non-trivial challenge. Solutions like proxy contracts and upgradeable smart contracts have been developed to address this issue.

4.Gas Limitations:- Ethereum requires users to pay for computational resources using a unit called "gas." Complex smart contracts with excessive computation can run into gas limitations, affecting their execution. Developers need to optimize gas usage to ensure cost-effective contract deployment.

Using Smart Contracts to Enforce Legal Contracts

Smart contracts have the potential to revolutionize the way legal contracts are created, executed, and enforced. Here's how they can be used in the context of legal agreements:

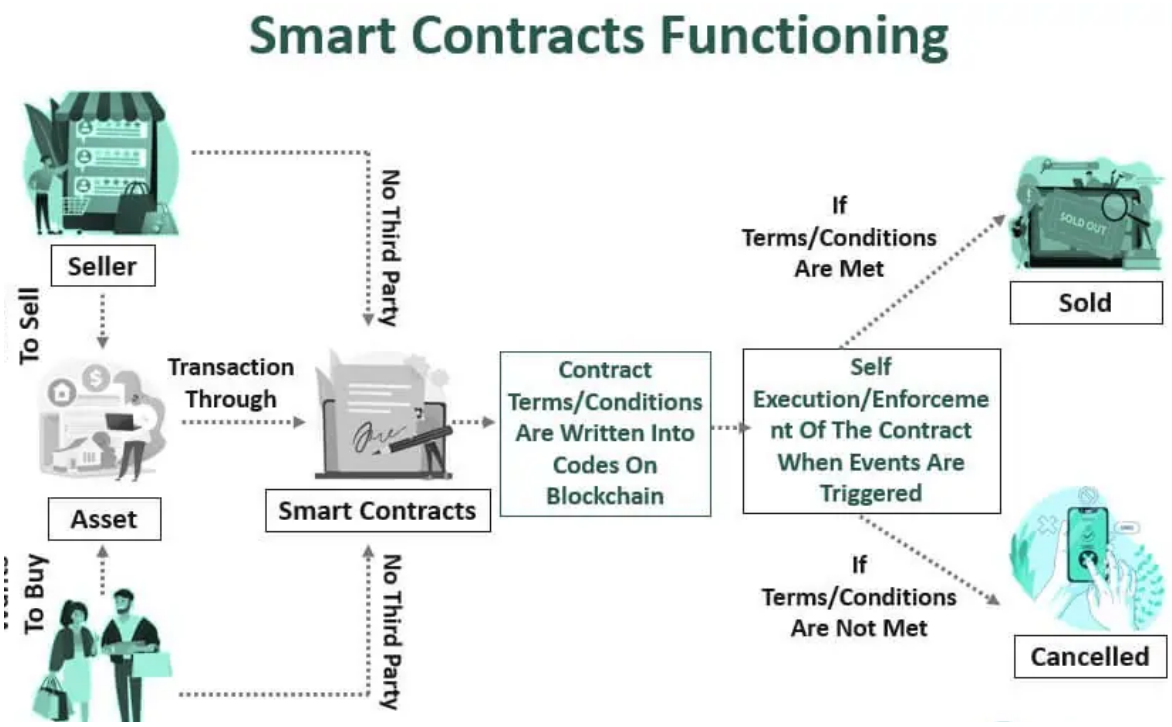


1.Automated Execution:- Smart contracts can automatically execute the terms of a legal agreement when predefined conditions are met. For example, in a supply chain contract, payment can be automatically released to a supplier when goods are received and verified.

2.Trustless Escrow:- Smart contracts can act as trustless escrow services. In a real estate transaction, for instance, the buyer can send funds to a smart contract, which holds them until the property title is transferred, ensuring both parties fulfill their obligations.

3.Transparency and Immutability:- The terms and execution of a legal contract on the blockchain are transparent and immutable. This provides a clear and tamper-proof record of all actions and transactions related to the contract, reducing disputes and the need for third-party arbitration.

4.Cost and Time Efficiency:- Smart contracts eliminate the need for intermediaries and manual paperwork, reducing legal and administrative costs. They also expedite the contract execution process, as it can be completed as soon as conditions are met.

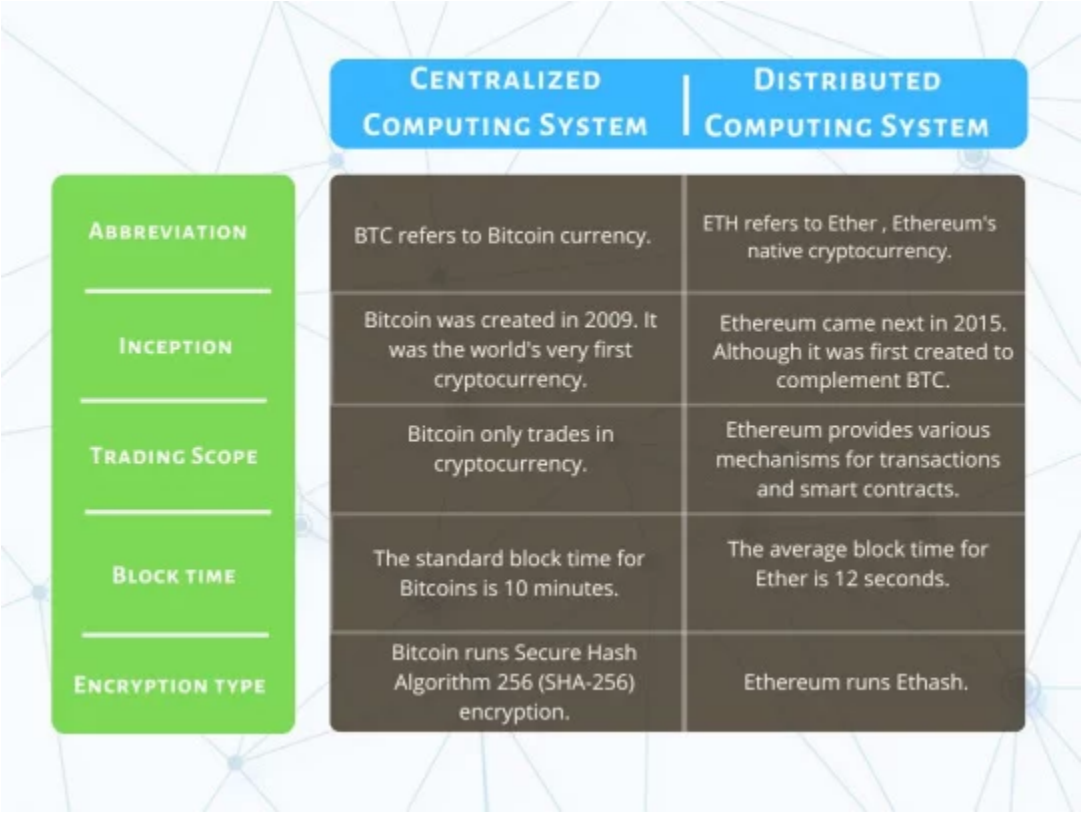


Comparing Bitcoin Scripting vs. Ethereum Smart Contracts

While both Bitcoin scripting and Ethereum smart contracts are programmable, they serve different purposes and have distinct characteristics:

Bitcoin Scripting:

- Bitcoin scripting is primarily designed for financial transactions and offers a more limited set of functionalities compared to Ethereum.
- It allows for basic operations like multi-signature wallets, time-locked transactions, and conditional spending.
- Bitcoin scripting is not Turing complete, meaning it cannot express arbitrary computations or complex logic.
- It is more focused on transferring and securing value.



	Centralized Computing System	Distributed Computing System
Abbreviation	BTC refers to Bitcoin currency.	ETH refers to Ether , Ethereum's native cryptocurrency.
Inception	Bitcoin was created in 2009. It was the world's very first cryptocurrency.	Ethereum came next in 2015. Although it was first created to complement BTC.
Trading Scope	Bitcoin only trades in cryptocurrency.	Ethereum provides various mechanisms for transactions and smart contracts.
Block time	The standard block time for Bitcoins is 10 minutes.	The average block time for Ether is 12 seconds.
Encryption type	Bitcoin runs Secure Hash Algorithm 256 (SHA-256) encryption.	Ethereum runs Ethash.

Ethereum Smart Contracts:

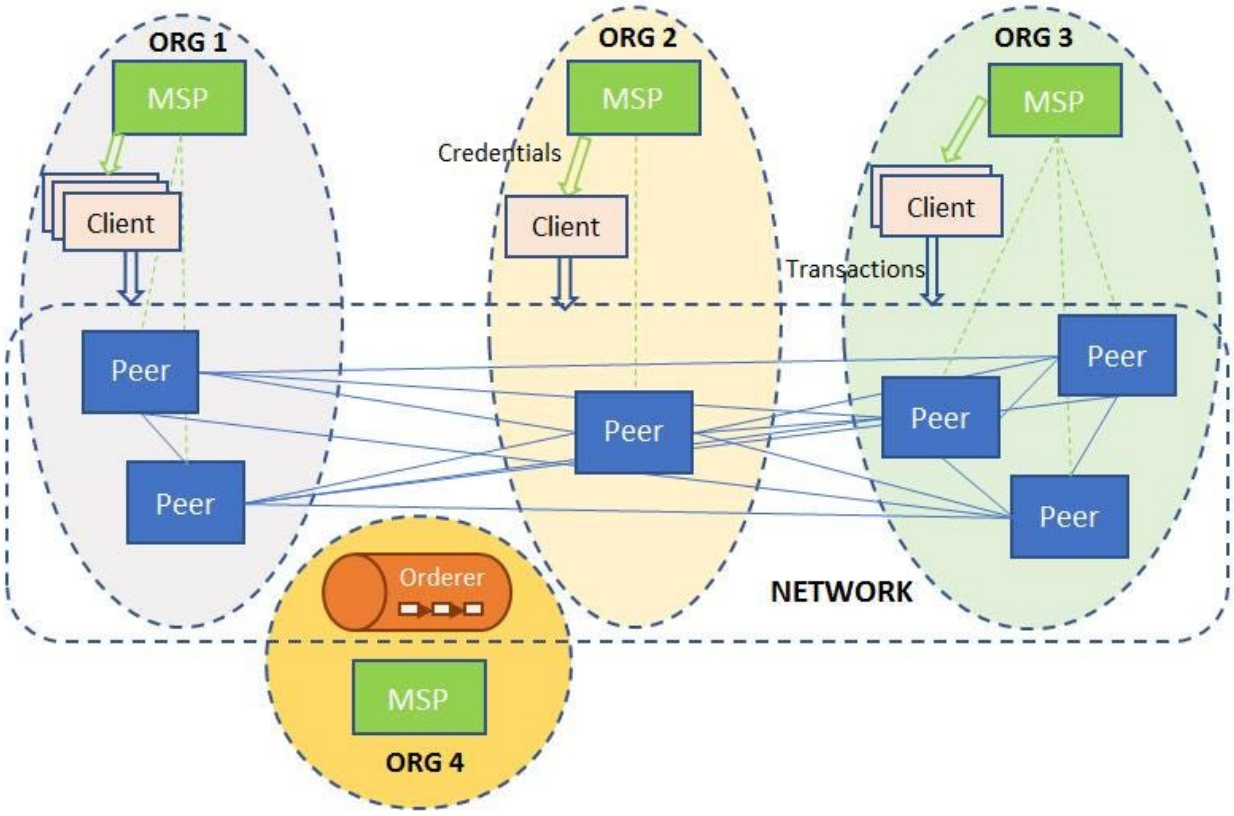
- Ethereum's smart contracts are Turing complete, enabling developers to create complex applications and automate a wide range of processes.
- They can be used for a vast array of use cases beyond finance, including decentralized applications (DApps), decentralized finance (DeFi), and decentralized autonomous organizations (DAOs).
- Smart contracts on Ethereum can interact with each other and with external data sources, allowing for more versatile and complex applications.

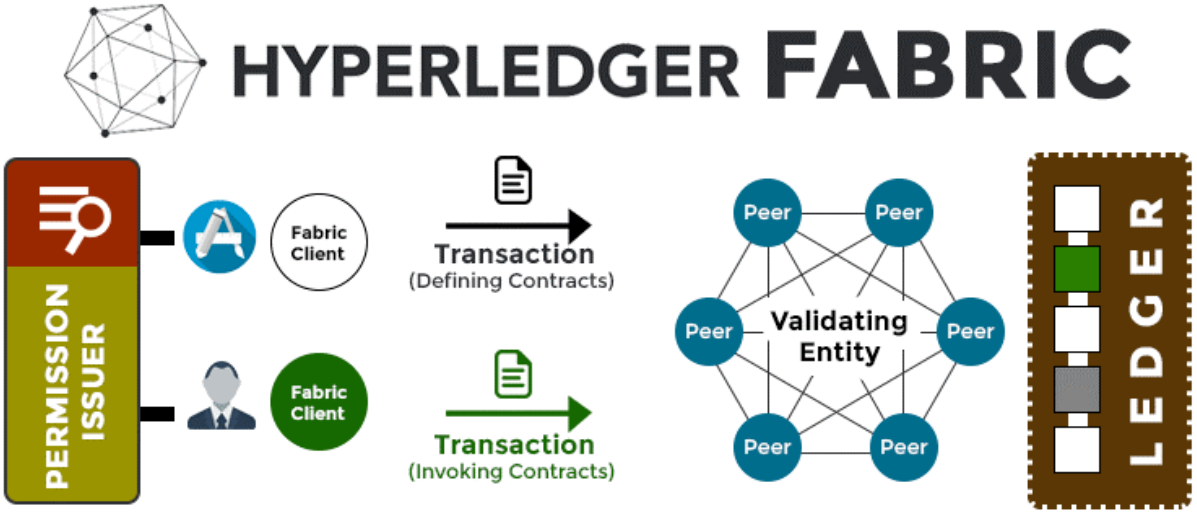
- They introduce a higher degree of programmability and flexibility compared to Bitcoin scripting.

Ethereum's introduction of smart contracts represents a significant advancement in blockchain technology, allowing for the creation of decentralized applications and the automation of complex processes. While the Turing completeness of smart contract languages like Solidity provides immense power, it also poses challenges related to verification and security. Smart contracts have the potential to disrupt traditional legal agreements by automating execution and ensuring transparency and trustlessness. Comparatively, Bitcoin scripting is more limited in scope, primarily serving as a means of transferring and securing value.

Hyperledger Fabric

Hyperledger Fabric is an open-source, permissioned blockchain platform developed under the Linux Foundation's Hyperledger project. Unlike public blockchains like Bitcoin and Ethereum, Fabric is designed for enterprise use cases, providing a versatile and modular framework for building blockchain networks. It offers several key features that distinguish it from other blockchain platforms:





The Plug-and-Play Platform

One of the standout features of Hyperledger Fabric is its plug-and-play architecture. This architecture allows organizations to tailor their blockchain networks to their specific needs by selecting and configuring the components that best suit their use cases. The main components include:

1.Membership Services:- Fabric uses a membership service provider (MSP) to manage identities and permissions. Organizations can define their own MSPs and customize them to align with their security requirements.

2.Consensus Mechanisms:- Fabric offers pluggable consensus mechanisms, allowing organizations to choose the consensus algorithm that best fits their network. This flexibility is vital for different use cases, from financial transactions to supply chain management.

3.Smart Contracts:- Fabric supports smart contracts (also known as "chaincode"), which are written in familiar programming languages like Go, Node.js, or Java. This versatility makes it easier for developers to create and integrate business logic into their blockchain applications.

4.Data Storage:- Fabric supports multiple database options, enabling organizations to use the database that suits their needs, such as LevelDB or CouchDB.

5.Privacy and Confidentiality:- Fabric is designed to support private and confidential transactions, making it well-suited for enterprise scenarios where data privacy is crucial.

Security Considerations and Best Practices for Fabric Networks

Security is a paramount concern in enterprise blockchain deployments. Hyperledger Fabric incorporates several security features and best practices to ensure the integrity and privacy of data and transactions:

1.Identity Management:- Fabric uses a robust identity management system through MSPs. Each participant in the network has a unique digital identity, and access controls are enforced through policies defined in MSPs.

2.Private Data Collections:- Fabric allows for private data collections where sensitive information can be shared only among authorized parties. This ensures that confidential data remains private while still benefiting from the security and immutability of the blockchain.

3.Endorsement Policies:- Smart contracts in Fabric are subject to endorsement policies that determine which participants must approve a transaction for it to be considered valid. This adds an additional layer of security and control.

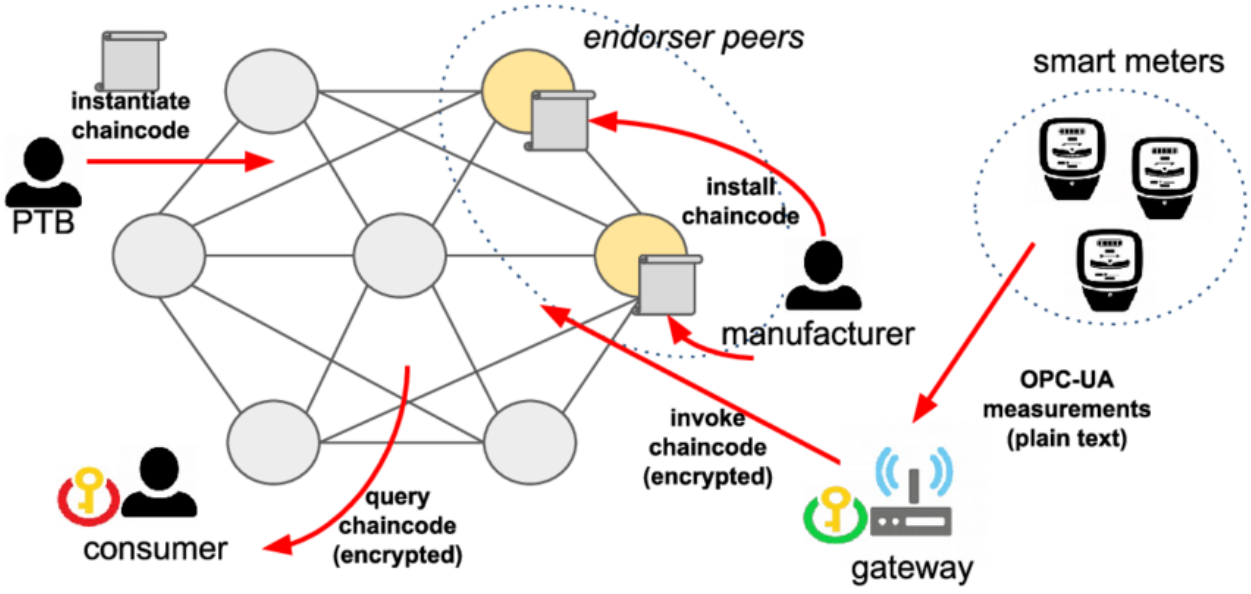
4.Immutable Ledger:- Once data is written to the Fabric blockchain, it cannot be altered or deleted, providing a tamper-resistant history of transactions.

5.Channel Isolation:- Fabric supports the creation of multiple channels, each acting as a separate blockchain network. This allows organizations to isolate and control the flow of data between specific parties.

6.Cryptography:- Fabric employs cryptographic techniques to secure data in transit and at rest. Data is encrypted during transmission using Transport Layer Security (TLS) and can be encrypted on the storage layer.

Secure Communication and Encryption in Fabric Networks

Secure communication and encryption are essential aspects of Hyperledger Fabric's design to protect sensitive information:



1.Transport Layer Security (TLS):- Fabric uses TLS encryption to secure communication between nodes in the network. This ensures that data transmitted over the network is encrypted and protected from eavesdropping and interception.

2. End-to-End Encryption:- Fabric supports end-to-end encryption for data at rest. This means that data stored on the blockchain or in private data collections can be encrypted to prevent unauthorized access, even within the network.

3. Private Data Collections:;- When using private data collections, data is shared selectively and only among authorized participants, further enhancing data privacy and confidentiality.

4. Zero-Knowledge Proofs:- Fabric is designed to support zero-knowledge proofs, which allow parties to prove that they possess certain information without revealing the actual data. This is particularly useful for privacy-preserving applications.

Hyperledger Fabric is a versatile and secure blockchain platform designed for enterprise use cases. Its plug-and-play architecture allows organizations to customize their blockchain networks, and its focus on security includes robust identity management, private data collections, endorsement policies, and cryptographic techniques. Fabric ensures secure communication through TLS encryption, supports end-to-end encryption for data at rest, and offers mechanisms like zero-knowledge proofs for enhanced data privacy and confidentiality. These features make Fabric a strong choice for organizations seeking to implement secure and flexible blockchain solutions.

Practice Questions

1. Purpose of Bitcoin's creation:

Bitcoin was created as a decentralized digital currency to enable peer-to-peer transactions without the need for intermediaries like banks. Its significance lies in providing a secure and transparent way to transfer value globally, as well as in pioneering blockchain technology.

2.Challenges faced by the Bitcoin network:

Bitcoin faces challenges such as scalability (limited transaction throughput), energy consumption (Proof of Work), regulatory scrutiny, and issues related to forks and consensus among miners.

3 Identify the scripting language:

Bitcoin uses a scripting language known as Bitcoin Script for creating transaction outputs and defining spending conditions.

4. Cryptography in blockchain technology:

Cryptography in blockchain ensures security by encrypting data, generating digital signatures for transactions, and enabling secure access control. It plays a vital role in data integrity, authentication, and privacy.

5.Turing completeness in smart contract languages:

Turing completeness means that a programming language or system can simulate a Turing machine, making it capable of solving a wide range of computational problems. Smart contract languages with Turing completeness are highly versatile.

6. Identify one advantage of using Bitcoin for financial transactions:

An advantage of using Bitcoin is its borderless nature, allowing for international transactions without the need for currency conversions or intermediaries. It also offers transparency and reduced transaction costs.

7. Limitations of the Bitcoin blockchain:
Limitations of the Bitcoin blockchain include scalability issues, slow transaction confirmation times, energy-intensive mining, lack of programmability, and limited privacy features.

8. Compare the scripting languages used in Bitcoin and Ethereum:

Bitcoin uses Bitcoin Script, which is primarily used for simple transactions and lacks complete Turing capability. Ethereum uses Solidity, a Turing-complete language, enabling complex smart contracts.

9. Bitcoin transactions security and safety:

Bitcoin transactions use cryptographic signatures to verify ownership and secure the transfer of funds. They are recorded in a decentralized and immutable ledger, making it difficult to alter or manipulate transaction history.

10. The concept of a smart contract :

Smart contracts are self-executing contracts with the terms of the agreement directly written into code. On Ethereum, they automate contract execution, eliminating the need for intermediaries and ensuring trustless enforcement of agreements.

11.Advantages of smart contracts and Ethereum:

Advantages include automation, trustlessness, transparency, reduced costs, and global accessibility. Ethereum's programmable nature allows for diverse applications beyond simple currency transfers.

12. Limitations of using Hyperledger Fabric:

Hyperledger Fabric is typically used in private or consortium networks. Limitations for public use include scalability, as it's designed for smaller, trusted networks, and the complexity of setting up and managing a Fabric network.

13. Advantage of using Ethereum smart contracts in supply chain management:

Ethereum smart contracts can improve supply chain management by providing transparency and traceability. For example, they can track the movement of goods, automate payment settlements, and reduce fraud.

14. The role of a peer in Hyperledger Fabric :

A peer in Hyperledger Fabric maintains the ledger and executes smart contracts. It acts as a node that stores and processes transactions, maintaining a copy of the ledger. A diagram would illustrate its role within the network.

15. Explain the use of smart contracts for peer-to-peer lending:

Smart contracts can automate the lending process by defining loan terms, interest rates, and repayment conditions. Funds are held in escrow, and the contract enforces repayments, reducing the need for intermediaries.

16. The ethical implications of using blockchain technology in various industries:

Ethical implications include privacy concerns, data ownership, environmental impact (e.g., energy consumption of PoW), potential misuse

(e.g., illegal activities on anonymous networks), and the need for responsible governance.

17. The concept of a channel in Hyperledger Fabric:

A channel in Hyperledger Fabric is a private communication path between a subset of network participants. It allows for confidential transactions and data segregation, ensuring that only authorized parties can access specific data.

18.Hyperledger Fabric handles data privacy:

Hyperledger Fabric ensures data privacy by using channels, where different parties can have their private transactions. For example, in a supply chain network, suppliers and buyers can have a private channel for sensitive trade data while sharing some information on a public channel.

19. The security implications of a peer-to-peer network:

Peer-to-peer networks offer increased decentralization and resistance to single points of failure but may be vulnerable to malicious peers. Centralized networks provide greater control but can be targeted for attacks and data breaches.

20. The advantages of consensus mechanisms in various blockchain networks:

Common consensus mechanisms include Proof of Work (e.g., Bitcoin) and Proof of Stake (e.g., Ethereum 2.0). Advantages include security, decentralization, and trust. Diagrams can depict the validation process.

21.Identity management features in Hyperledger Fabric:

Hyperledger Fabric offers more robust identity management with Certificate Authorities (CAs) and Membership Service Providers (MSPs). This

enhances security and privacy by ensuring that only authorized participants can access the network.