Introduction to Blockchain

1. Introduction to Blockchain

Blockchain is a decentralized and distributed digital ledger technology that records transactions across multiple computers in a secure and tamper-proof manner. Each block in the chain contains a list of transactions, and these blocks are linked together cryptographically. Blockchain is widely used in cryptocurrencies, smart contracts, and decentralized applications (DApps) due to its transparency, security, and immutability.

Key Characteristics of Blockchain:
- **Decentralization:** No central authority controls the network.
- **Immutability:** Once a block is added, it cannot be altered.
- **Transparency:** Transactions are visible to all participants.
- **Security:** Uses cryptographic algorithms for verification.

2. Blockchain Terminology

- **Block:** A container of transactions, linked to the previous block.
- **Ledger:** A digital record of transactions stored in blocks.
- **Node:** A participant in the blockchain network.
- **Consensus Mechanism:** The method used to validate transactions and add blocks.
- **Public Key & Private Key:** Cryptographic keys used for securing transactions.
- **Smart Contract:** Self-executing contracts with predefined rules.
- **Miner:** A participant who validates transactions and adds them to the blockchain.

3. Types of Blockchain

Public Blockchain

- **Definition:** A decentralized, open network where anyone can participate, validate transactions, and create smart contracts.
- **Examples:** Bitcoin, Ethereum
- **Use Cases:** Cryptocurrencies, decentralized finance (DeFi), public smart contracts
- **Pros:** Transparent, secure, trustless, censorship-resistant
- **Cons:** Slower transactions, high energy consumption (Proof of Work), scalability issues

2. Private Blockchain (Permissioned)

- **Definition:** A blockchain controlled by a single entity or a consortium, where only authorized participants can join and validate transactions.
- **Examples:** Hyperledger Fabric, R3 Corda

- **Use Cases:** Enterprise applications, supply chain management, banking, and corporate solutions
- **Pros:** Faster transactions, controlled access, better scalability
- **Cons:** Centralized control, reduced trust, not fully decentralized

3. Consortium Blockchain (Federated)

- **Definition:** A hybrid model where multiple organizations share control over the blockchain network instead of a single entity.
- **Examples:** Energy Web Foundation, Quorum by J.P. Morgan
- **Use Cases:** Interbank settlements, supply chain, healthcare, trade finance
- **Pros:** More decentralized than private blockchains, enhances trust among partners, efficient for B2B transactions
- **Cons:** Requires trust among consortium members, less transparent than public blockchains

4. Hybrid Blockchain

- **Definition:** A combination of public and private blockchains, where certain data is open to the public while some remain restricted.
- **Examples:** Dragonchain, XinFin (XDC)
- **Use Cases:** Financial services, supply chain, government applications
- **Pros:** Balances privacy and transparency, customizable security
- **Cons:** Complex to implement, requires governance

4. Types of Nodes in Blockchain

Nodes are the backbone of blockchain networks, helping maintain decentralization, security, and consensus. Here are the different **types of nodes in blockchain**:

---

1. Full Nodes

- **Definition:** These nodes store a complete copy of the blockchain ledger and validate transactions independently.
- **Function:** Enforces consensus rules, verifies transactions, and maintains network integrity.
- **Examples:** Bitcoin Core, Geth (Ethereum)
- **Pros:** High security, decentralization, trustless validation
- **Cons:** Requires significant storage and computational power

---

2. Light Nodes (SPV Nodes - Simplified Payment Verification)

- **Definition:** Light nodes store only block headers instead of the full blockchain.
- **Function:** They rely on full nodes to verify transactions, making them lightweight and fast.
- **Examples:** MetaMask (Ethereum), Electrum (Bitcoin)
- **Pros:** Fast synchronization, low storage requirement, suitable for mobile and lightweight wallets
- **Cons:** Depend on full nodes for transaction validation, less secure

---

3. Mining Nodes

- **Definition:** Nodes that participate in proof-of-work (PoW) mining by solving cryptographic puzzles.
- **Function:** They validate transactions and add new blocks to the blockchain.
- **Examples:** Bitcoin miners, Ethereum miners (before Ethereum 2.0)
- **Pros:** Secure the network, enable new block creation
- **Cons:** High energy consumption, requires specialized hardware (ASICs, GPUs)

---

5. Consensus Mechanisms

Consensus mechanisms are protocols that help blockchain networks agree on the validity of transactions without relying on a central authority. They ensure security, decentralization, and immutability.

a) Proof of Work (PoW)
- Used by Bitcoin.
- Miners solve complex mathematical puzzles to validate transactions.
- Energy-intensive and slow but secure.

b) Proof of Stake (PoS)
- Used by Ethereum 2.0.
- Validators are chosen based on the number of coins they hold.
- More energy-efficient than PoW.

c) Delegated Proof of Stake (DPoS)
- Voting-based consensus.
- Used by EOS and TRON.

d) Practical Byzantine Fault Tolerance (PBFT)
- Used in private blockchains like Hyperledger Fabric.

- Faster than PoW and PoS but requires trust among participants.

6. Blockchain Platforms

a) Bitcoin
- The first cryptocurrency based on blockchain.
- Uses PoW for consensus.

b) Ethereum
- Supports smart contracts and decentralized applications (DApps).
- Uses PoS for consensus.

c) Hyperledger Fabric
- Private and permissioned blockchain.
- Used in enterprise solutions.

d) Binance Smart Chain (BSC)
- Designed for fast and efficient transactions.
- Uses a modified PoS mechanism.

e) Corda
- Designed for financial institutions.
- Uses a permissioned blockchain model.

7. What is Hashing?

A **hash function** takes an input (transaction, block data, etc.) and produces a unique, fixed-length string (hash).
Example using **SHA-256** (used in Bitcoin):

🔹 **Input:** "Blockchain"
🔹 **Output (Hash):**
`5d41402abc4b2a76b9719d911017c592`

Properties of Hashing:

✅ **Deterministic** → Same input always gives the same output.
✅ **Fast Computation** → Hash generation is quick.
✅ **Irreversible** → Cannot retrieve the original data from the hash.
✅ **Collision-Resistant** → No two different inputs produce the same hash.
✅ **Small Changes = Big Differences** → A slight change in input drastically changes the hash.

Hashing in Blockchain

A. Block Hashing

Each blockchain block has a **hash**, linking it to the previous block.
🔷 If data in any block changes, the hash changes, making tampering nearly impossible.

B. Merkle Tree (Transaction Hashing)

Blockchain transactions are stored in a **Merkle tree**, where:

- Transactions are **hashed**.
- Pairs of hashes are combined to form a **Merkle Root**.
- The Merkle Root is stored in the block header.

SHA-256 (Secure Hash Algorithm 256-bit)

SHA-256 is a widely used cryptographic hash function developed by the NSA.

a) Features:
- Produces a 256-bit (32-byte) hash value.
- Used in Bitcoin and other cryptocurrencies.
- Highly secure and resistant to collisions.

b) Example of SHA-256:
```
Input: "Blockchain"
Output: 92c9b5db31f75b6a3e1a88b4c0e3eeaa4f5021bdf4c2cfaf1d4166b5c40413a7
```
SHA-256 (Secure Hash Algorithm 256-bit) is a cryptographic hash function widely used in blockchain for ensuring security, data integrity, and immutability. It converts any input into a fixed **256-bit (64-character)** hash, making it irreversible and collision-resistant. The process involves **converting input to binary, padding the message**, splitting it into **512-bit blocks**, and performing **64 rounds of bitwise operations and modular arithmetic** to generate the final hash. In **Bitcoin mining**, SHA-256 is used to create unique block hashes by adjusting a **nonce** until a valid hash is found, ensuring **proof-of-work (PoW) consensus**. It also plays a crucial role in **transaction hashing and Merkle Tree construction**, enabling efficient and tamper-proof transaction verification. Due to its **one-way nature, security, and speed**, SHA-256 remains the backbone of blockchain security, making it nearly impossible to alter past transactions without re-mining the entire chain.

8. Merkle Trees in Blockchain

A **Merkle Tree** (or **Hash Tree**) is a cryptographic data structure used in blockchain to efficiently and securely verify large sets of transactions. It ensures **data integrity, quick verification, and tamper-proof storage**.

What is a Merkle Tree?

A Merkle Tree is a **hierarchical structure** where:

- Each **leaf node** contains the **hash** of a transaction.
- Each **non-leaf node** is the hash of its two child nodes.
- The **Merkle Root** is the final single hash at the top of the tree.

🔷 **Purpose:** Provides a compact, tamper-proof summary of all transactions in a block.

---

2. How Merkle Trees Work

Step-by-Step Process

1. **Hash each transaction** → Convert raw transaction data into a hash.
2. **Pair up hashes** and hash them together → Forms the next level of the tree.
3. **Repeat until only one hash remains** → This final hash is the **Merkle Root**.
4. **Store the Merkle Root in the Block Header** → Secures the integrity of all transactions.

Benefits of Merkle Trees in Blockchain

✅ **Efficient Verification:** Allows quick proof that a transaction is part of a block.
✅ **Security & Integrity:** Any change in a transaction **alters the Merkle Root**, making fraud impossible.
✅ **Scalability:** Instead of storing all transactions, only the **Merkle Root** is stored in the block header.
✅ **Efficient SPV (Simplified Payment Verification):** Light nodes can verify transactions without storing the full blockchain.

What is Proof of Inclusion?

- It allows a **light node** (e.g., a mobile wallet) to verify that a transaction exists in a block **without storing the entire blockchain**.

- The full node provides a **Merkle Proof**, which consists of:
  - The **transaction hash** (Tx)
  - The **Merkle Root** of the block
  - A **Merkle Path** (hashes of sibling nodes needed to reconstruct the root)

If the light node can reconstruct the **Merkle Root** using the provided hashes, the transaction is verified.

9. Digital Signatures in Blockchain

A **digital signature** is a cryptographic technique used to authenticate transactions and messages in a blockchain. It ensures **data integrity, authenticity, and non-repudiation** without requiring centralized verification.

---

1. How Digital Signatures Work

Digital signatures use **asymmetric cryptography** (public and private key pairs) to sign and verify data.

Process:

1. **Key Generation:** A user generates a public-private key pair using cryptographic algorithms (e.g., RSA, ECC, ECDSA).
2. **Signing:** The sender hashes the message and encrypts it using their private key.
3. **Verification:** The receiver decrypts the hash using the sender's public key and compares it with a new hash of the received message.

If both hashes match, the message is **authentic** and **unaltered**.

---

2. Properties of Digital Signatures

✅ **Authenticity:** Ensures the message came from the legitimate sender.
✅ **Integrity:** Prevents tampering; even a small change invalidates the signature.
✅ **Non-repudiation:** The sender cannot deny sending the message.

---

3. Digital Signature Algorithms Used in Blockchain (covered as a part of Assignment 1)

Algorithm	Used By	Pros	Cons
ECDSA (Elliptic Curve Digital Signature Algorithm)	Bitcoin, Ethereum	High security, lower key size	More complex math
EdDSA (Edwards-Curve Digital Signature Algorithm)	Cardano, Monero	Faster verification, resistant to side-channel attacks	Not widely adopted
RSA (Rivest-Shamir-Adleman)	Traditional cryptography	Well-established, widely used	Large key sizes, slower than ECC
Schnorr Signatures	Bitcoin Taproot	Aggregation of multiple signatures, efficiency	Requires adoption in blockchains

10. Commitments in Blockchain

Commitments are cryptographic tools used to lock in a value while keeping it hidden until a later stage.

a) Types of Commitments:

- Time-Locked Commitments: Require a certain time period before they are revealed.
- Merkle Commitments: Use Merkle trees to structure data efficiently.

b) Use Cases of Commitments:

- Ensuring security in smart contracts.
- Protecting privacy in zero-knowledge proofs.
- Enhancing security in multi-party computations.

Conclusion

Blockchain is a revolutionary technology with numerous applications beyond cryptocurrencies. Key concepts such as digital signatures, hashing, SHA-256, and consensus mechanisms play a crucial role in ensuring the security, integrity, and decentralization of blockchain networks. As the field evolves, new developments in cryptographic security and blockchain platforms will continue to shape the future of digital transactions and decentralized systems.