Blockchain Technology SoS Reading Project

Pagoti Hemanth Naidu - 210050117
Mentor: Shikhar Agrawal

may 2023

Contents

0.1 Introduction

Bitcoin was introduced in 2009 by Satoshi Nakamoto in response to the 2008 global financial crisis and the need for a decentralized, peer-to-peer electronic cash system. Nakamoto's vision was to create a currency free from central control, censorship-resistant, and capable of secure, borderless transactions.

0.1.1 Demand & Supply of Bitcoin

Bitcoin's supply is regulated by Satoshi Nakamoto's algorithm, limiting the total number of bitcoins that can ever exist to 21 million. The demand for bitcoin is determined by market forces and influenced by factors such as investor sentiment and adoption. Balancing supply and demand is essential for maintaining stability in the Bitcoin market and mitigating the risk of inflation.

0.1.2 Double Spend Problem

Satoshi Nakamoto's design of Bitcoin ensures that money can only be spent once through the concept of ownership and peer-to-peer networks. With public keys, everyone can know how much each owner possesses while maintaining anonymity. Hash functions are used for encoding transactions and mining purposes, while proof of work enables Bitcoin to function as a cryptocurrency. The tamper-proof nature of blockchain ensures that once transactions are added, they cannot be modified, and each block in the blockchain contains the hash of the previous block. Smart contracts are self-executing contracts with the terms of the agreement directly written into code, automating and enforcing the agreed-upon conditions without the need for intermediaries. They enable secure and transparent transactions in various applications, from finance to supply chain management.

0.2 Motivation

0.2.1 Blockchains in High Level

- Blockchain utilizes a tamperproof data structure that ensures data integrity and security.
- Starting with a genesis block, each subsequent block in the chain holds information in the form of bitstrings or binary representations.
- Once data is added to a block and appended to the chain, it becomes immutable and cannot be removed or altered.
- This feature of blockchain, along with its decentralized and distributed nature enables various applications such as cryptocurrency, smart contracts and trust.

0.2.2 Applications

- **Governance**: Land records, health records, transportation data, virtual currency, electronic wills, passport, identification, etc.
- **Commercial**: supply chains, auctions, gaming, sale of music, financial services, smart grid etc.
- **Disruptive**: Cryptocurrencies, Initial Coin Offerings

0.3 cryptocurrency

Bitcoin has no central trudtrd authority, cryptography brings in trust and peer to peer networks provide decentralization. Bitcoin satisfies the characteristics of acceptability, portability, durability, divisibility, and fungibility.

- **Acceptability**: Bitcoin is widely accepted as a form of payment and store of value.
- **Portability**: Bitcoin can be easily transferred and accessed across geographical boundaries.
- **Durability**: Bitcoin's digital nature makes it resistant to physical damage or deterioration.
- **Divisibility**: Bitcoin is divisible into small units, allowing for precise transactions of any value.
- **Fungibility**: Bitcoin units are interchangeable, meaning that each unit holds the same value and can be exchanged without distinction.

Each transaction is brodcasted among all so that double spend is avoided.
Consensus protocols are mechanisms used in blockchain networks to achieve agreement among participants on the validity of transactions and the order in which they are added to the blockchain and this is also know as proof of work.

0.4 Peer to Peer

A permissionless, distributed system allows an arbitrary number of participants to join or leave at any time, enabling seamless peer-to-peer transactions where individuals have the freedom to pay anyone within the network, fostering inclusivity and decentralization.

0.4.1 Consensus Protocol

- To establish consensus in a protocol, cryptography is employed to secure transactions and validate the integrity of the data.

- While a peer-to-peer network enables direct communication and information sharing among participants.
- The proof-of-work mechanism adds a layer of computational effort to validate transactions and prevent malicious behavior, and Merkle trees provide an efficient and verifiable way to store and verify the integrity of large sets of data within the blockchain.
- Together, these components form the foundation for a robust and trustless consensus protocol.

0.4.2 Popular P2P versions

Napster

Napster was one of the earliest peer-to-peer networks that facilitated file sharing. It used a centralized server to maintain an index of available files, allowing users to search and download files directly from each other, enabling widespread file sharing but faced legal challenges due to copyright concerns.

Gnutella

Gnutella is a decentralized peer-to-peer network where participating nodes connect directly with each other. It operates on a query flooding mechanism, where search queries propagate across the network, and peers respond with matching files, ensuring a distributed and scalable file sharing system.

Distributed Hash Table

DHT is a decentralized peer-to-peer network architecture that provides efficient lookup and storage of key-value pairs. It operates by partitioning the hash space among participating nodes, enabling direct lookup and storage of data based on keys. DHTs provide fault tolerance, scalability, and efficient resource utilization in distributed systems and are commonly used in applications like BitTorrent and distributed storage systems.

0.5 Hash functions

Each block in the blockchain contains hash of the previous block, the hash is recognised as an unique ID.

0.5.1 Basic properties of Hash functions

- Input to the hash function can be of any length.
- Output of the hash function should be of fixed size.
- Hah should be efficiently computed.

0.5.2 Random Oracle

It is like a black box that takes input and generates a random output based on that input. It provides a consistent and unpredictable response to any query. Random oracles are often used to model idealized cryptographic functions and serve as a building block for designing secure protocols and systems.
However, in practice, real-world hash functions are used as a substitute for random oracles.

0.5.3 Properties of Cryptographic Hash functions

Collision Resistence

Collision resistance, in the context of cryptographic hash functions, means that it is computationally infeasible to find two different inputs that produce the same hash output. It ensures that a small change in the input will result in a significantly different hash value, making it difficult to forge or manipulate data without detection.
The birthday paradox states that in a group of relatively few people, the probability of two individuals sharing the same birthday is surprisingly high.

Hiding

suppose r takes values from r_1, r_2, r_3, ..., r_n with probability $p(r_1)$, $p(r_2)$, $p(r_3)$, ..., $p(r_n)$.
The minimum entropy is defined as follows

$$min - entropy = min_{i=1}^{n} - log(x_i)$$

where a secret value r is chosen from a probability distribution with high min-entropy and combined with another value x to compute the hash h(x||r), it is infeasible to find the original value of x given only the hash h(x||r).

Puzzle Friendliness

Hash function is puzzle friendly if for every N bit output Y if K is choosen from a probability distribution with high minimum entropy, then it is infeasible to find X such that

$$H(K||X) = Y$$

in time significantly less than $O(2^N)$.

0.5.4 Proof of Work

- Miner selects pending transactions.
- Miner combines hash of previous block, nonce, and transactions.

- Nonce is of 16 bits, miner can include any number of transactions.
- Miner adjusts nonce until the resulting block hash meets the threshold for network consensus, creating the next block.
- Once a valid block is created it should be brodcasted to other miners for verification.

0.6 SHA256 and Merkle Trees

Here's a high-level overview of how SHA-256 works:

- **Padding**: The input message is padded with length of message to ensure it meets certain requirements for the SHA-256 algorithm. Padding includes adding bits to the message to make it a multiple of 512 bits.
- **Initialization**: An Initialization Vector (IV) is a predetermined set of initial values used to initialize the compression process in certain compression algorithms, ensuring consistent and reproducible compression results for the same input data.
- **Processing**: The padded message is divided into blocks of 512 bits, and the algorithm processes each block sequentially. The processing involves a series of logical and bitwise operations, including message expansion, data mixing, and bitwise operations such as AND, OR, and XOR.
- **Compression**: Each block is compressed using a combination of logical and bitwise operations, updating the state of the hash values.
- **Finalization**: Once all the blocks have been processed, the resulting hash value is obtained by concatenating the updated hash values.

Miners use Merkle trees (also known as Merkle hash trees) as an efficient and secure way to summarize a large set of transactions within a blockchain. These trees play a crucial role in the creation of the next block in the blockchain. Let's walk through the process:

- Miners gather valid transactions and sort them. They hash each transaction individually and combine the hashed values in pairs until a single root hash, known as the Merkle root, remains at the top of the tree.
- The Merkle root is included in the block header along with other block information. Miners attempt to find a nonce value that, when hashed with the block header, produces a hash that is less than the threshold.

0.6.1 Advantages of Merkle Trees

- Merkle trees enable efficient verification and tamper detection, making it difficult for a miner to modify a specific transaction without changing the Merkle root.
- If a miner want to add or Modify a transaction while solving the puzzle it can be done in O(log(M)) operations where M is no of transactions included.
- Light clients(New commer of Bitcoin) require fewer resources, enabling users with limited capabilities to participate in blockchain networks.

0.6.2 Prooving to Light Client

There is a situation that A is a light clinet and B need to proove that one of his transaction happend to C to A then B can provide A with a compact proof known as a Merkle proof.
This proof consists of the transaction, its corresponding Merkle path, and the Merkle root. A can then verify the validity of the transaction by using the Merkle proof to trace the transaction's inclusion in the Merkle tree and comparing the computed Merkle root with the one stored by A.

0.6.3 Full Client using Merkle Trees

If a full client want to discard any of the transaction he stored he can still proove that remaing transactions belonging to the same merkle tree which may not be happend if he stores in a naive way i.e concatenating all the transactions.

0.7 Bitcoin Transactions

0.7.1 Block Header contents

The block header in a blockchain typically contains the following elements:

- **Previous Block Hash**: Hash of the preceding block in the blockchain.
- **Merkle Root**: Hash of all the transactions included in the block.
- **Timestamp**: The time when the block was created, which is used after to find appropriate threshold.
- **Nonce**: A number used in the mining process to find a valid block hash.
- **threshold**: 4 Bytes are used to represnt threshold also known as bits, to save space instead of storing 256 bits.

$$bits = b_1 b_2 b_3 b_4$$

$$Threshold = b2_2 b_3 b_4 * 256^{b_1 - 3}$$

0.7.2 Digital Signatures

Digital Signatures are hard to forge, easy to create and easy to verify.
These signatures use the below three alogorithms to complete the need:

Generation of shared, public key pair

This uses GenerateKeys algorithm which takes key size as the parameter and generates a key pair in which shared key is kept secret and public key is known to everyone.

sign the message

This uses Sign algorithm which takes message and shared key as the algorithm and encypts the message, in this case message is typically a transaction.

verify the signature

This uses Verify algorithm which takes the message, signature and cipher text and verify the signature, this is done by everyone to verify transactions.

0.7.3 ECDSA

Elliptic Curve Digital Signature Algorithm is a cryptographic algorithm used for generating and verifying digital signatures, in which public key is 512 bits, shared key is 256 bits, signature is 512 bits and the message should be 256 bits. Hash the message using SHA256 as many of the messages are not of 256 bits.

0.8 Transactions