CISC 322

Conceptual Report: Bitcoin Core

Friday, February 17, 2022

Group 36:

Victor Ghosh (18vg5@queensu.ca)

Ethan Davis (19ead6@queensu.ca)

Stefan D'Ippolito (19sadi@queensu.ca)

David Shen (19ds71@queensu.ca)

Table of Contents

Abstract

Through the breakdown of the conceptual architecture of a system we can provide a broad idea as to what the structure of a system is intended to be. This may vary significantly from the actual product in question due to short cuts, changes, and improvements but is useful none the less. As is the case with many systems, Bitcoin Core's architecture is not a strictly described concept. Instead, the following conceptual architecture analysis derives from a variety of sources from both developers and the wider community.

Introduction

Beginning in 2009, Bitcoin became a revolutionary new form of crypto currency. The idea was to create a form of monetary transaction that did not rely on any central bank or authority to verify purchases. This was achieved through the use of Blockchain technology and incentivized Bitcoin "mining" that would maintain the security of transactions. Thanks to this mining process the current Bitcoin blockchain now contains over 750,000 blocks, each one further securing the block before it. Since its creation the network has expanded significantly and includes a variety of different types of nodes. The following paper intends to deconstruct the conceptual architecture of a reference client node, also known as Bitcoin Core. We will discuss its components, uses, data flow, version history, and development all with a focus on the software's greater architectural style.

Derivation Process

Our research began with exploring the fundamental elements of the Bitcoin network, including its blockchain, mempool, and peer-to-peer network, to develop a thorough understanding of their interrelationships. We collaborated closely to pinpoint the key subsystems and their interactions. Through this Derivation Process, we conducted an in-depth analysis of the conceptual architecture of the Bitcoin Core network, which provided valuable insights into its security and functionality. By the end of our investigation, we had a comprehensive understanding of the Bitcoin Core network's structure and operations.

Architecture Style

Through our research, we discovered that the system is built on a peer-to-peer network, which is achieved using a blockchain, a distributed ledger that records all transactions. The architecture is designed to be transparent, secure, and resistant to manipulation, with incentives for network participants to maintain its integrity. The use of cryptography ensures the authenticity and privacy of transactions. Overall, the Bitcoin system's architecture emphasizes decentralization, which sets it apart from traditional centralized financial systems.

Conceptual Architecture

Overview

In the following conceptual architecture breakdown, we will consider the components present in a reference client, or bitcoin core, node. This node is assumed to be connected to the larger network in a traditional peer-to-peer fashion rather than utilizing a relay server.

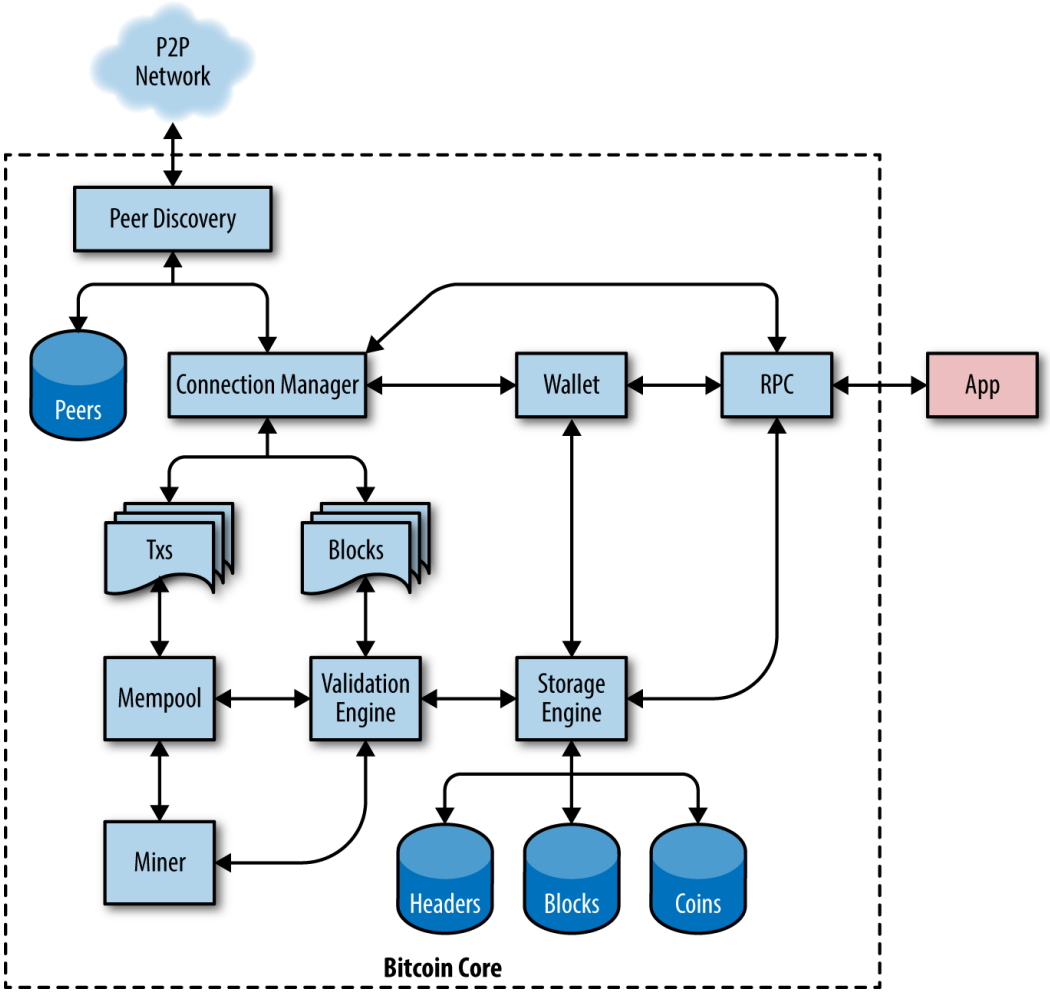

Figure 1: Diagram of Bitcoin Core Conceptual Architecture

(Cyberpunks-core – Github)

Subsystems Breakdown

Peer Discovery

The system will maintain a small list of known peers in order to contribute to the network and receive blockchain updates but when a node is first turned on this list will naturally be empty. The peer discovery component solves this by making a series of DNS queries to find a stable running node and open a TCP connection. It will send this node its IP address to be forwarded to other running Bitcoin systems, thus creating more connections. This process will continue until the peers list has been filled. At this point, the node has found an acceptable number of neighbors and will only search again if instructed by the connection manager because an existing peer has become inactive. In this case, the node will forward its address to its existing running neighbors rather than relying on a DNS server. Naturally, to maintain the peer discover functionality across the network, if this node is forwarded to an address by one of its neighbors it will forward it to the other peers. If the system happens to currently be looking for a new peer, it may also respond to the IP address itself.

Connection Manager

The connection, or address, manager works closely with peer discovery to maintain the list of peers. This component will keep the addresses found through peer discovery in memory and periodically move them to storage. It is also tasked with ensuring that no attacker can fill this table with nefarious connections. Lastly, the connection manager maintains a time stamp keeping track of the last time each peer was successfully communicated with. When a neighboring node remains inactive for too long the connection will be removed from the peers list and replaced by peer discovery.

Wallet

Bitcoin Core was originally designed, and still comes with, what is called a type-0 nondeterministic wallet. This refers to a collection of randomly generated private keys which can be used to sign transactions on the bitcoin blockchain. The original architecture called for 100 of these keys to be produced followed by more on an as needed basis. However, it is worth noting that a deterministic wallet, where the keys are generated from one master key, is now recommended as a replacement. Regardless of which type is used, the system will need to access the correct key from the wallet when authorizing transactions.

RPC

Bitcoin Core utilizes RPC (remote procedure protocol) as a communication protocol between nodes in a network. The RPC interface enables programs and scripts to interact with Bitcoin Core, with functionalities ranging from spending electronic wallets to accessing private data and conducting other operations that may involve loss of money, data, or privacy.

Transactions

A transaction refers to the record of value transfer from one user to another. Transactions require a cryptographic signature generated by the sender using their private key, which proves that they are the legitimate owner of the inputs being spent. The digital signature is then validated using the sender's public key.

Mempool

The mempool (short for "memory pool") is a Bitcoin node component that stores unconfirmed transactions that have been transmitted across the network. Once a Bitcoin transaction is broadcast, nodes add it to their mempool while waiting for it to be confirmed by the network. As new transactions are received, they are added to the mempool, where they await confirmation by miners. Miners validate transactions and include them in new blocks in the blockchain.

Miner

Miners are network participants who validate transactions and add them to the blockchain. When a new Bitcoin transaction is broadcasted, it is added to the mempool, where it waits for a miner to include it in a new block. Miners compete to be the first to validate a new block in the blockchain by solving complex mathematical problems.

Validation Engine

The Bitcoin Core validation engine is a critical component of the Bitcoin network. It is responsible for verifying that transactions and blocks added to the blockchain are valid according to the Bitcoin protocol. The validation engine employs a combination of cryptographic techniques, consensus rules, and computational puzzles to check the validity of each transaction and block.

Digital signatures are used to ensure that only the rightful owner of a particular bitcoin address can spend the funds held in that address, while UTXOs are used to prevent overspending and to maintain the integrity of the blockchain. When a user sends a transaction, it must reference one or more UTXOs as inputs, which are then consumed and converted into new UTXOs that are sent to the recipient addresses. The validation engine checks that the transaction spends only valid UTXOs and that the sum of the inputs is greater than or equal to the sum of the outputs, preventing overspending and ensuring that the transaction is valid.

Proof of work is another key technique employed by the validation engine to secure the network against attacks and ensure that new blocks are added to the blockchain in a decentralized and trustless manner. The proof of work algorithm requires miners to solve a complex computational puzzle, which helps to prevent malicious actors from easily adding invalid blocks to the blockchain. By requiring a significant amount of computational effort to add new blocks, proof of work ensures that the Bitcoin network remains secure and resistant to attack.

In addition to digital signatures, UTXOs, and proof of work, the Bitcoin Core validation engine enforces a set of consensus rules that dictate how the Bitcoin protocol operates. For example, there are limits on the size of each block, which helps to prevent network congestion and maintain the efficiency of the network. The validation engine checks that new blocks and transactions meet these consensus rules and rejects any that do not, ensuring that the network operates in a secure and reliable manner. The validation engine plays a critical role in maintaining the decentralized, secure nature of the Bitcoin network.

Storage Engine

The storage engine in Bitcoin Core is responsible for storing and managing the blockchain data, this includes information about all the transactions that have ever occurred on the Bitcoin network which is the blockchain. The blockchain data is organized into various components, including blocks, headers, and coins.

Blocks

Each block in the blockchain is a collection of transactions that have been verified and added to the blockchain. Each block includes a header that contains metadata about the block, including block height, timestamp, and the hash of the previous block in the chain. The header is used to link blocks together, forming the blockchain. The storage engine in Bitcoin Core maintains a database of blocks, which allows nodes to verify the integrity of the blockchain and retrieve information about specific transactions.

Headers

The headers in the blockchain are organized into a separate database table, which allows for more efficient retrieval of header information. The header table includes information about each block in the blockchain, such as the block height, timestamp, and the hash of the block. The headers are used to verify the validity of the blockchain and determine the longest valid chain.

Coins

The coins in Bitcoin Core are represented by unspent transaction outputs (UTXO). Each UTXO represents a specific amount of bitcoin that has been sent to an address but has not yet been redeemed. The storage engine in Bitcoin Core maintains a database of UTXOs, which allows nodes to verify transactions and determine the available balance of a particular address. The UTXO database includes information about the transaction output, such as the amount of bitcoin, the script that unlocks the output, and the block in which the transaction was included.

The storage engine in Bitcoin Core uses a LevelDB database to store the blockchain data. LevelDB is a key-value store that allows for efficient storage and retrieval of large amounts of data. It is optimized for high write throughput and efficient disk usage, which is important for the storage of the constantly growing blockchain. The LevelDB database used by Bitcoin Core is organized into several different tables, including tables for blocks, headers, and UTXOs, as well as tables for other components of the software, such as wallets and mempool data. The storage engine in Bitcoin Core is a critical component of the software, as it is responsible for maintaining the integrity and security of blockchain data. It allows nodes to store and retrieve the data efficiently and provides the foundation for the various functions of the software.

Use Cases

Transaction

The following use cases follow a bitcoin transaction from start to finish. First a Peer wants to transfer a certain number of bitcoin. The Peer Discovery component is notified, and the system will maintain a small list of known peers to contribute to the network. The Connection Manager will then keep the addresses found through peer discovery in memory and periodically move them to storage. Bitcoin core's Wallet software specifies the recipient's Bitcoin address and the amount of Bitcoin they want to send. The user's wallet signs off on the transaction using the private key associated with each input to the transaction. This is done to prove that this is the owner of the bitcoin being sent. Once signed off the transaction is created and broadcast across the network through the Connection Manager. Nodes pick up transaction and check if it is valid through the Validation Engine. If the transaction is valid it is added to the Mempool to wait until it is confirmed by a miner. The Miner takes transactions from the Mempool, recursively hashes pairs of transaction hash values until there is only one remaining, which is called the Merkle Root. This along with timestamp, nBits, nonce and a reference to the previous block hash is placed in the block header. For a transaction to be added to a block in the chain the miner needs to solve a difficult math puzzle based on a cryptographic hash algorithm to show proof that the miner spent significant computing effort (also called proof of work or PoW). After the Miner solves the PoW algorithm the last step is the independent validation of each new block by every node on the network through the Validation Engine. Once it is verified this information is stored in the Storage Engine and is included in a block and recorded on the blockchain. This information is then sent to the RPC and shown on their interface to the receiver. The recipients Bitcoin wallet software sees this transaction on the network, so it adds the received bitcoin to their wallet.

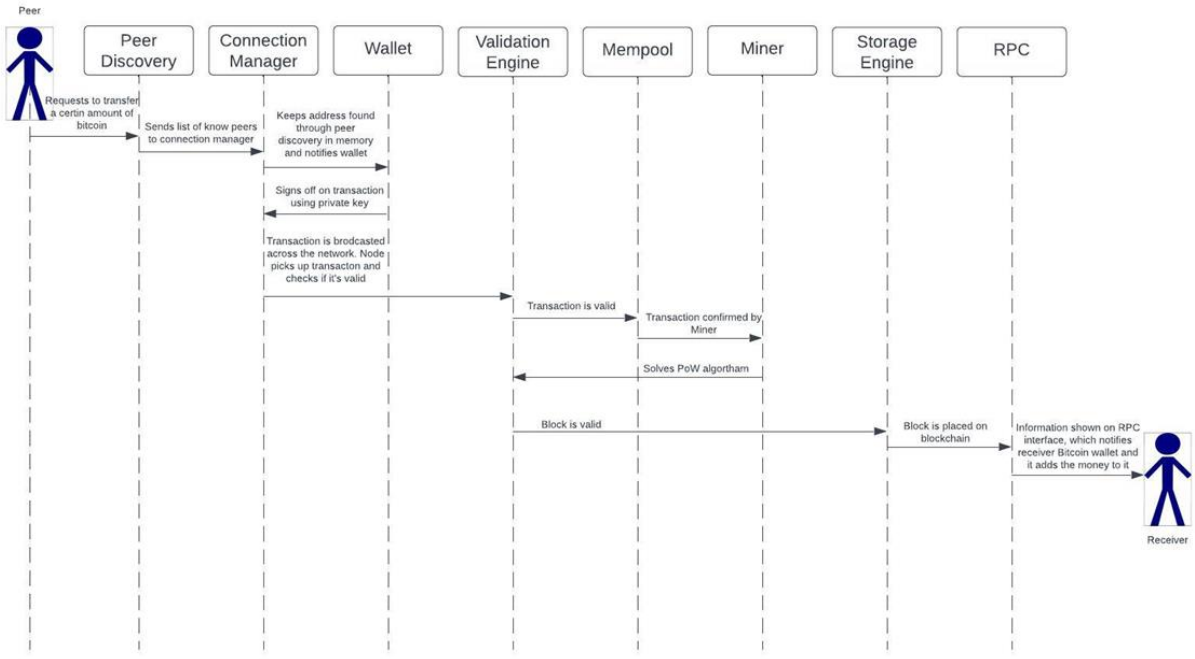

Figure 2: Sequence Diagram for Use Case: Transaction

Mining

The second use case goes through the process of adding transactions to the bitcoin blockchain. This use case assumes that the node we are viewing is the “winner” of this blocks proof-of-work competition. The node is constantly receiving new transactions from its peer network and retransmitting these transactions to the other neighbors. When a new block is received it validates it and ends its computation on the previous block. It will also slot this new block into its copy of the blockchain. The node then compares the transactions, throwing out any transactions in its memory pool that were included in the new block, before creating the header and constructing the candidate block. The candidate is sent to the miner which will compute the nonce and return the hashed block. It will also send this block to storage engine so it may be added to the nodes running copy of the block chain and the reward keys can be added to its wallet. Finally, the newfound block is sent to all the node’s peers.

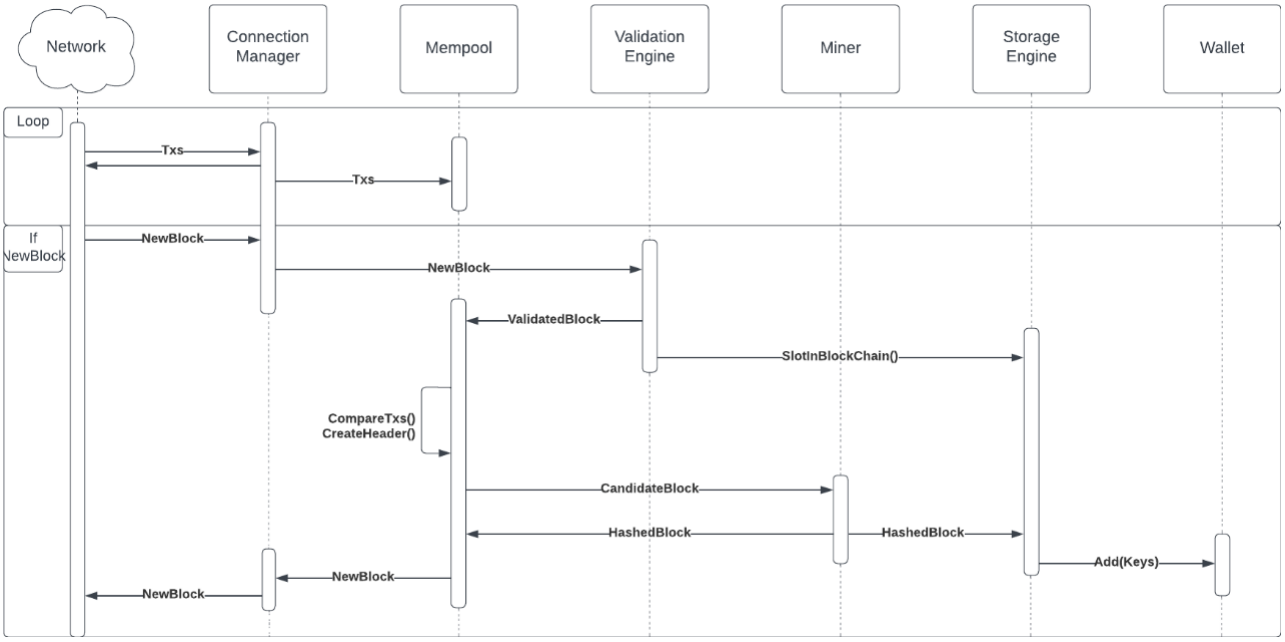

Figure 3: Sequence diagram for Use Case: Mining

Data Flow

Bitcoin Core is a decentralized system that uses a peer-to-peer network to maintain a shared ledger of all transactions on the Bitcoin blockchain. The data flow in Bitcoin Core can be described as follows:

1. **Transaction creation**: When a user creates a new transaction, it is broadcast to the network as a message that contains information about the sender, the recipient, and the amount being sent.

2. **Transaction propagation**: The transaction message is propagated through the network of Bitcoin nodes, which consists of computers running the Bitcoin Core software. Each node that receives the transaction message verifies its validity before relaying it to its neighbors.

3. **Mining**: Miners compete to add new transactions to the Bitcoin blockchain by solving complex mathematical puzzles. The first miner to solve the puzzle is rewarded with a block subsidy and the fees associated with the transactions in that block. This process is known as mining.

4. **Block Reproduction/Propagation**: Once a block has been mined, it is propagated through the network as a message that contains information about the transactions in the block. Nodes that receive the block message verify its validity and add it to their local copy of the blockchain.

5. **Validation**: Each node in the network independently validates the blockchain to ensure that it is consistent with the rules of the Bitcoin protocol. This includes verifying the signatures on each transaction, checking for double-spending, and confirming that the total supply of Bitcoin is not being exceeded.

Bitcoin Core's data flow is vastly complex, with transactions and blocks being propagated through a network of nodes that work together to maintain the integrity and security of the blockchain.

Control Flow

The control flow of Bitcoin Core can be described as the set of processes and rules that govern how transactions and blocks are validated and added to the blockchain. The control flow is based on a set of consensus rules that are defined by the Bitcoin protocol, and which are enforced by the network of Bitcoin nodes running the Bitcoin Core software. The control flow is as follows:

1. **Transaction validation**: When a new transaction is received by a node, it is validated to ensure that it meets the rules of the Bitcoin protocol. This includes checking that the transaction has a valid signature, that it does not double-spend any inputs, and that the sum of inputs and outputs is balanced. If the transaction is invalid, it is rejected and not propagated further.

2. **Transaction relay**: If a transaction is valid, it is relayed to other nodes on the network. Nodes that receive the transaction perform their own validation checks, and if the transaction is valid, they relay it to their own neighbors.

3. **Block validation**: Like the data flow, the control flow also has validation. When a miner creates a new block, it must validate all the transactions in the block to ensure that they meet the rules of the

Bitcoin protocol. This includes checking that the transactions are valid, that the total size of the block does not exceed the maximum allowed, and that the block contains valid proof of work.

4. Block Reproduction/Propagation: When a miner has created a valid block, it is propagated through the network. Nodes that receive the block validate it, and if it is valid, they add it to their copy of the blockchain.

5. Conclusions: To the fact that Bitcoin Core uses a P2P architecture style this means that there is no central authority that can determine which version of the blockchain is authoritative. Instead, consensus is achieved through a process called Proof of Work, which requires miners to compete to solve mathematical puzzles to add new blocks to the chain. The longest chain with the most proof-of-work is considered the authoritative version of the blockchain.

6. Reorganization: Occasionally, two miners may find valid blocks at the same time, leading to a temporary fork in the blockchain. In this case, the network will eventually converge on the longest chain, and the shorter chain will be discarded. This is known as a blockchain reorganization.

Bitcoin Core has a strong yet complex control flow, with transactions and blocks being validated by a network of peers. The control flow is based on a set of consensus rules that are defined by the Bitcoin protocol, and which are enforced by the network of Bitcoin peers running the Bitcoin Core software.

Version Control

Bitcoin Core is the reference implementation of the Bitcoin protocol and is the software that runs the Bitcoin network. As a result, it is critical to the stability and security of the network. Over the years, the software has gone through several major updates, with each new version introducing new features, performance improvements, and bug fixes. Below are the more significant updates and releases of Bitcoin Core over the years.

Bitcoin Core 0.1.0: This was the initial release of the Bitcoin software, created by Satoshi Nakamoto. It included basic transaction functionality and the ability to mine and validate blocks.

Bitcoin Core 0.3.21: This release was the first major update to the Bitcoin Core software and included several important improvements to the network's functionality. It introduced the ability to send and receive bitcoin over the network, which allowed users to transact with each other without needing a third-party intermediary. It also added support for more operating systems.

Bitcoin Core 0.8.0: This release introduced a new block chain format that improved security and efficiency by reducing the amount of disk space required to store the block chain. It also added support for multiple block chains, Also, it included a new memory pool implementation that improved transaction handling and reduced the likelihood of transaction "spamming" attacks.

Bitcoin Core 0.10.0: This release added dynamic fee estimation, which made it easier for users to estimate the appropriate fee to include with their transactions to ensure they were processed quickly. Also added support for hierarchical deterministic wallets, which allowed users to more easily manage and backup their wallet data, It achieved this by randomizing the order in which inputs and outputs were listed in transactions.

Bitcoin Core 0.12.0: This release included the Segregated Witness (SegWit) update, which improved the network's capacity and reduced the vulnerability to transaction malleability attacks. In turn reduced the size of transactions and increased the number of transactions that could be included in each block.

Bitcoin Core 0.15.0: This release introduced a new fee estimation algorithm that improved transaction confirmation times, and support for hierarchical deterministic key generation, which made it easier to manage multiple wallets. It also introduced a new wallet format that made it easier to backup and recover wallet data.

Bitcoin Core 0.16.0: This release was focused on improving the usability of the software. It introduced support for bech32 addresses, which are easier to read and more efficient than previous address formats.

Bitcoin Core 0.18.0: This release included several improvements designed to improve privacy and transaction handling. It added support for better privacy features, such as the ability to create new addresses for each transaction and the ability to use Tor to help protect privacy. It also introduced support for the Lightning Network, a second-layer payment protocol that allows for faster, cheaper, and more private transactions.

Bitcoin Core 0.21.0: This release included several notable changes that improved the functionality and security of the software. It added better support for Tor hidden services, which helps protect user privacy by hiding their location and identity.

Bitcoin Core is a continuously evolving software project that has undergone numerous updates over the years. These updates have added important new features, improved performance, and enhanced security.

Responsibilities Among Developers

Bitcoin Core is an open-source software project, it is maintained and developed by a community of volunteer developers who work on the codebase and contribute to its ongoing improvement. As an open-source project, there are no formal responsibilities assigned to individual developers. However, the community of developers who contribute to Bitcoin Core have the following responsibilities:

Code Builders: Are responsible for writing and submitting code changes to the Bitcoin Core codebase. This can include bug fixes, feature enhancements, and security patches.

Code Reviewers: Are responsible for reviewing code changes submitted by other developers. Code review is an important part of the development process to ensure that the code changes are safe, efficient, and consistent with the project's coding standards.

Code Testers: Are responsible for testing the Bitcoin Core software to ensure that it functions correctly and is free from bugs and other issues. This includes both manual testing and the development of automated testing tools.

Security: Are responsible for ensuring that the Bitcoin Core software is secure and resistant to attacks. This includes identifying and patching vulnerabilities, conducting security audits, and implementing best practices for secure software development.

The development of Bitcoin Core is a collaborative effort, and each developer contributes in their own way based on their skills and interests. The success of the project depends on the continued involvement and contributions of the community of developers who work on it.

Lessons Learned

Researching the conceptual architecture of Bitcoin Core offers several key lessons. One of the main takeaways is the importance of security and decentralization in the design of blockchain-based systems. Bitcoin's architecture emphasizes preventing attacks and ensuring the integrity of the system, through features such as cryptographic algorithms. The transparency and collaboration of the open-source community have played a significant role in the development and evolution of the Bitcoin system, and this approach has been adopted by many subsequent blockchain projects. Overall, researching the architecture of Bitcoin Core provides valuable insights into the principles and challenges of blockchain design and implementation.

Conclusion

The conceptual architecture of Bitcoin Core is a key factor in its success as a peer-to-peer electronic cash system. The system’s emphasis is on decentralization using blockchain and cryptographic protocols, which provides a secure alternative compared to a traditional centralized financial system. The cryptocurrency space has seen significant growth and evolution, with many new and exciting projects in recent years. As the cryptocurrency and blockchain space continues to evolve, the conceptual architecture of Bitcoin Core will serve as a foundation for the development of future systems and applications.

Data Dictionary

Proof-of-work

A number of data values which requires a significant set amount of computation to resolve

Coinbase transaction

The first transaction in every block to be paid to the miner. This is the only type of transaction with no payee.

Nondeterministic wallet

Also known as a JBOK (Just a Bunch Of Keys) wallet this wallet type contains a set of randomly generated keys to sign transactions on the blockchain

Deterministic wallet

A wallet in which each key was generated by using a master key as the seed. In this way the wallet can be fully rebuilt with the master key.

Reference client (Bitcoin Core)

A node containing a wallet, miner, network node, and full blockchain database

References

Antonopoulos, Andreas M. “Mastering Bitcoin, 2nd Edition.” *O'Reilly Online Learning*, O'Reilly Media, Inc., https://www.oreilly.com/library/view/mastering-bitcoin-2nd/9781491954379/ch03.html.

“Chapter 3: 'Bitcoin Core: The Reference Implementation'.” *Chapter 3: 'Bitcoin Core: The Reference Implementation' · GitBook*, https://cypherpunks-core.github.io/bitcoinbook/ch03.html.

Gr0kchain. “Understanding the Data behind Bitcoin Core.” *Bitcoin Developer Network*, Bitcoin Developer Network, 5 Apr. 2019, https://bitcoindev.network/understanding-the-data/.

“How to Access Bitcoin Mempool.” *QuickNode*, https://www.quicknode.com/guides/web3-sdks/how-to-access-bitcoin-mempool#:~:text=What%20is%20Bitcoin%20Mempool%3F,is%20known%20as%20the%20mempool.

“P2P Network.” *Bitcoin*, https://developer.bitcoin.org/reference/p2p_networking.html.

“A Peer-to-Peer Electronic Cash System.” *Bitcoin*, https://bitcoin.org/en/bitcoin-paper.

“Running a Full Node.” *Bitcoin*, https://bitcoin.org/en/full-node.

Spirkovski, Zoran. “Who Really Controls the Bitcoin Core Software?” *BeInCrypto*, 16 Dec. 2018, https://beincrypto.com/who-controls-bitcoin-core/.

TheBitcoin Application Itself Forwardinbitcoinimprovementproposals ... - Gi. https://dl.gi.de/bitstream/handle/20.500.12116/16570/DFN-Forum-Proceedings-001.pdf.

TheBitcoin Application Itself Forwardinbitcoinimprovementproposals ... - Gi. https://dl.gi.de/bitstream/handle/20.500.12116/16570/DFN-Forum-Proceedings-001.pdf.

“Transactions.” *Bitcoin*, https://developer.bitcoin.org/devguide/transactions.html.

“Understanding Bitcoin Core: The Reference Implementation.” *YouTube*, YouTube, 12 Aug. 2021, https://www.youtube.com/watch?v=wLYdcH37phE.