CISC 322/326 Assignment 2 Concrete Architecture of Bitcoin Wallets

March 19th, 2023

Group #3: IKUN - FAMILY
Jinyang Chen 19jc66@queensu.ca
Chiyu Wang 19cw33@queensu.ca
Gaoyuan Bao 19gb20@queensu.ca
Rundong Yu 18ry16@queensu.ca

Table of Contents

1 Abstract

In the previous report, the Bitcoin core system's conceptual architecture was examined.
The Bitcoin core's concrete architecture and its bitcoin wallet subsystem are examined in this report. We concentrate on its usability and security aspects. The architecture of the wallet is thoroughly examined, highlighting major components such as the user interface, transaction management, and key distribution modules. The wallet's security features, such as the use of cryptographic protocols, and secure storage of private keys.
For the wallet's usability aspects, such as compatibility with various operating systems and mobile devices, as well as integration with other services and applications.
Overall, the analysis demonstrates our comprehension of the concrete architecture and technical specifics of the bitcoin wallet and associated subsystems.

2 Introduction

In brief view, reflection analysis is a powerful tool for understanding the behavior of Bitcoin Core and improving its performance, reliability, and security. The use of concurrency is essential for the network to handle multiple transactions and users simultaneously, improving its efficiency and scalability. When it comes to Bitcoin wallets, the complex and multifaceted concrete architecture requires careful consideration of its components and security measures, such as multi-factor authentication and encryption of private keys, to protect users' funds against security threats. Finally, the concrete architecture analysis of the use cases "Making transactions" and "Mining" for the blockchain showed the components and processes involved in completing these tasks, highlighting the importance of verification and network transmission in ensuring the successful completion of transactions and mining requests.

3.1 reflection analysis:

Network Subsystem: Reflection analysis may be used to track how the Network Subsystem behaves when there is a high volume of transactions. It is easy to spot potential bottlenecks or congestion in the network by looking at the network traffic and evaluating latency,

throughput, and other metrics. To enhance the system's overall performance, this data can be utilized to improve routing algorithms, optimize network design, or boost bandwidth.

Consensus Subsystem: Reflection analysis may be used to track the Consensus Subsystem's activity during periods of heavy block validation. It is easy to spot potential performance problems or vulnerabilities in the consensus method by looking at the validation process and evaluating performance indicators like validation time, memory use, and CPU utilization. The consensus algorithm, validation speed, or system security may all be improved using this information.

Wallet Subsystem: Reflection analysis may be used to track each wallet's actions and spot any possible security risks. Anomaly behaviour pointing to a security breach or unauthorized access can be found by checking the transaction history, balance, and other metadata linked to each wallet. Security issues like wallet theft or malicious behaviour may be identified and remedied using this information.

RPC Subsystem: The RPC Subsystem's behaviour may be tracked via reflection analysis, which can also be used to spot any possible performance or security problems. It is feasible to locate potential bottlenecks or weaknesses in the RPC interface by monitoring performance indicators like request latency, response time, and resource use. The RPC subsystem may be optimized, response times can be increased, and security issues can be found and addressed.

Block Storage Subsystem: The Block Storage Subsystem, which manages to store and retrieve the blockchain data, may be seen via reflection analysis. Potential block storage performance issues may be found by monitoring metrics like disc I/O, read/write times, and memory utilization. This data may speed up read/write operations, decrease disc I/O, and optimize the block storage subsystem.

Mempool Subsystem: The Mempool Subsystem, which is in charge of handling the unconfirmed transactions that have been broadcast to the network, may be observed using reflection analysis. It is feasible to spot potential bottlenecks or inefficiencies in the mempool mechanism by monitoring data like mempool size, transaction acceptance rate, and transaction eviction rate. The mempool subsystem can be optimized, the transaction acceptance rate can be raised, or the size of the mempool may be decreased using this information.

P2P Subsystem: Reflection analysis may be used to monitor the P2P Subsystem's actions, which create and maintain connections between network nodes. P2P network problems may be detected by tracking metrics like connection count, peer discovery rate, and message throughput. The P2P Subsystem may be optimized, connection reliability can be increased, and message latency can be decreased using this information.

Mining Subsystem: The Mining Subsystem, which creates new blocks for the blockchain, may be observed in action via reflection analysis. It is easy to spot potential performance problems or vulnerabilities in the mining algorithm by monitoring hash rate, block production time, and difficulty adjustment rate. The mining subsystem can be optimized, block creation times can be sped up, and system security can be raised using this data.

3.2 Concurrency

The Bitcoin Core software's support for concurrency is crucial since it allows numerous processes to be completed at once, enhancing the system's effectiveness and performance. In Bitcoin Core, concurrency is utilized in the following contexts:

Transaction Processing: To manage several transactions at once, the Network Subsystem employs a combination of parallelism and asynchronous processing. Each node may independently check and verify a new transaction since it is handled asynchronously by several nodes in parallel when it is received. This increases the transaction processing's overall speed and effectiveness.

Block Validation: The Consensus Subsystem employs parallel validation to expedite the block validation procedure. Several nodes validate a new block in parallel when it is received. Each node separately verifies the block; if it is legitimate, it is put into the blockchain. This simultaneous validation method ensures the speedy and effective certification of fresh blocks.

Mining: Several miners compete to crack the cryptographic puzzle needed to build a new block under the concurrent mining model of Bitcoin Core. The reward is given to the first miner to solve the challenge after each miner has worked on a distinct block in parallel. With this concurrency, new blocks are made rapidly and effectively.

Wallet management: The Bitcoin Core Wallet Subsystem is made to manage several wallets at once. Users may make numerous wallets and carry out simultaneous transactions on each one without interfering with the others. To prevent transactions on one wallet from impacting the balances or transactions on another, each wallet uses a set of unique key pairs to achieve this goal.

RPC Interface: The Bitcoin Core RPC Subsystem has a multi-threaded architecture to handle several requests simultaneously. Each request is dealt with on a distinct thread via the RPC interface, which may simultaneously handle numerous requests from outside apps. Because of this concurrency, the system may process a high volume of submissions without slowing down or going down.

Peer-to-Peer Communication: The Network Subsystem of Bitcoin Core uses a peer-to-peer protocol to allow nodes to connect. This protocol's concurrent handling of many connections enables nodes to communicate with many other nodes.

UTXO Management: Bitcoin Core uses unspent Transaction Output (UTXO) modelling to control the blockchain's status. The UTXOs of a transaction are added to the UTXO set, which is kept in memory after it is verified. The UTXO set is managed using concurrent data structures in Bitcoin Core, allowing for the conflict-free processing of several transactions simultaneously.

Script Validation: The criteria under which a transaction can be spent are specified by Bitcoin Core using a programming language. The scripting engine validates several trades at once, increasing validation's effectiveness.

Block Propagation: To enhance the likelihood that a new block will be added to the blockchain, a miner must swiftly distribute new blocks to other nodes on the network. The block propagation process is optimized by Bitcoin Core using contemporary approaches, such as parallelizing block data encoding and decoding.

Network Peering: Network peering is a technique Bitcoin Core uses to create connections between nodes on the network. Nodes may connect to many other nodes at once thanks to network peering, which is intended to manage numerous connection attempts concurrently.

4. Bitcoin wallet analysis

Section 1: Components of Bitcoin Wallet Architecture

The concrete architecture of a Bitcoin wallet consists of several key components, including private keys, public addresses, wallet software, user interfaces, security measures, and network connectivity. Private keys are secret codes that allow users to control their Bitcoin funds, while public addresses are unique identifiers that allow users to receive Bitcoin transactions. The wallet software is responsible for managing a user's private keys and public addresses, while the user interface provides a visual interface for users to interact with their funds. Security measures like multi-factor authentication, backup and recovery options, and encryption of private keys are essential for protecting users' funds. Finally, network connectivity is necessary for the wallet software to communicate with the Bitcoin network and send and receive transactions.

Section 2: Core Wallet Functionality

The core wallet functionality of a Bitcoin wallet is responsible for the underlying functionality of the software, including private key management, transaction signing, and communication with the Bitcoin network. The core wallet is typically implemented in a low-level language like C++ for maximum performance and security. It is essential that the core

wallet is designed with robust security measures in place, such as encryption of private keys and multi-factor authentication.

One key aspect of the core wallet functionality is private key management. Private keys are the secret codes that allow users to control their Bitcoin funds, and are therefore the most critical aspect of Bitcoin wallet security. To ensure maximum security, private keys are typically stored in a digital "wallet" that is encrypted and protected with a password. The wallet software is responsible for managing the private keys, generating new addresses for receiving transactions, and signing transactions to send Bitcoin.

Transaction signing is another critical aspect of the core wallet functionality. When a user wants to send Bitcoin, the wallet software generates a transaction that specifies the amount of Bitcoin to be sent and the recipient's public address. The wallet then uses the user's private key to sign the transaction, ensuring that only the user with the corresponding private key can authorize the transaction.

Finally, the core wallet functionality is responsible for communication with the Bitcoin network. The wallet software communicates with the network to broadcast transactions, receive updates on the status of transactions, and synchronize the user's balance with the network. It is essential that the communication with the Bitcoin network is secure and tamper-proof, to prevent unauthorized access to the user's funds.

Section 3: User Interface Design

The user interface of a Bitcoin wallet is responsible for providing an easy-to-use interface for users to interact with their funds. The user interface should provide users with a clear picture of their funds and transaction history, as well as options to send and receive transactions. The user interface can be implemented as a desktop or mobile application, a web-based interface, or as part of a hardware wallet. It is essential that the user interface is intuitive and easy to use, while still providing robust security features like multi-factor authentication and backup and recovery options.

The user interface should provide users with a clear and concise overview of their Bitcoin balance and transaction history. This can be achieved through the use of graphs, charts, and other visual aids that help users understand their transaction history and balance trends. Additionally, the user interface should provide easy access to the key functionalities of the wallet, such as sending and receiving transactions, managing private keys, and setting up security options. The user interface should be designed with a focus on user experience, ensuring that users can easily navigate the software and complete transactions quickly and efficiently.

In addition to ease of use, the user interface must also be designed with robust security measures in mind. Multi-factor authentication, such as two-factor authentication (2FA), can help to prevent unauthorized access to the wallet, while backup and recovery options like seed phrases can help users recover their funds in the event of lost or stolen private keys. The

user interface should also be designed to protect against phishing attacks and other security threats, with clear warnings and notifications when suspicious activity is detected.

Section 4: Security Measures

Security is perhaps the most critical aspect of Bitcoin wallet design, as the loss or theft of private keys can result in the permanent loss of the user's funds. Bitcoin wallets must be designed with robust security measures in place to protect against a wide range of security threats, including hacking, malware, phishing, and physical theft.

One of the most effective security measures for Bitcoin wallets is multi-factor authentication (MFA). MFA requires users to provide multiple forms of identification, such as a password and a fingerprint scan, before accessing their funds. This can help to prevent unauthorized access to the wallet, even if a hacker or attacker gains access to the user's password.

Encryption of private keys is another critical security measure for Bitcoin wallets. Private keys must be protected with strong encryption algorithms to prevent unauthorized access, and users must be advised to keep their private keys safe and secure. Backup and recovery options, such as seed phrases, can help users recover their funds in the event of lost or stolen private keys.

Hardware wallets are another option for users who require maximum security for their Bitcoin funds. Hardware wallets are physical devices that store private keys offline, making them immune to hacking and malware attacks. Hardware wallets can also be designed with additional security measures, such as PIN codes and biometric authentication, to provide an extra layer of protection for users' funds.

5.1 Use Case 1

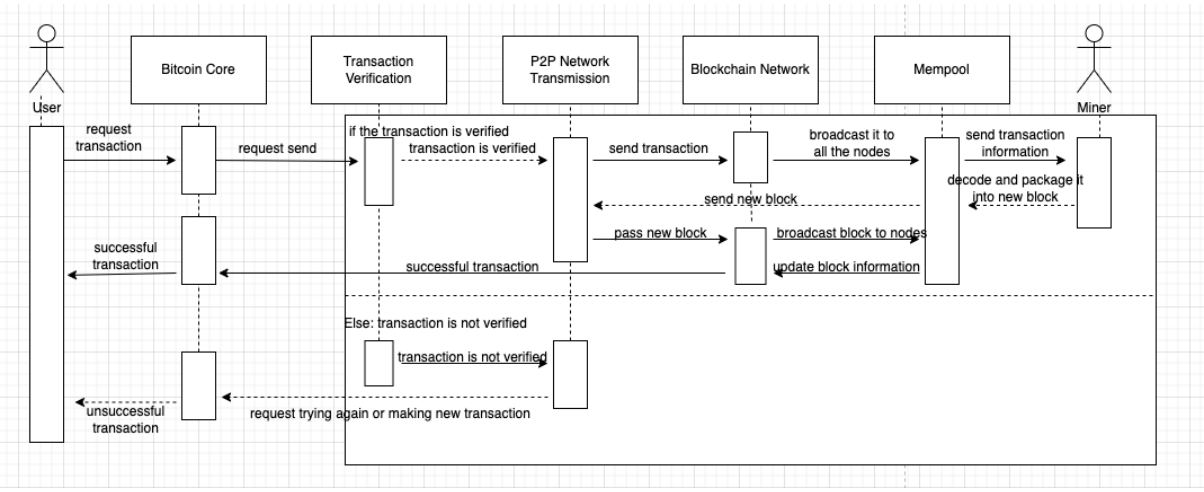

Figure 1. *Use Case 1: Making Transactions*

For concrete architecture analysis of use case: transaction, there are six main components (other than the user), which are Bitcoin core (the interface/app that interacts with the user), Transaction verification (verifies the regularity, legitimacy and validity of transactions), P2P Network Transmission (responsible for communicating with the blockchain and They are Bitcoin core (the interface/app that interacts with the user), transaction verification (verifying that transactions are regular, legitimate and valid), P2P Network Transmission (responsible for communicating with the blockchain and transferring data), Blockchain Network (storing and managing blockchain data and communicating with all nodes within the block), and Miner (decoding transaction information and packaging it into a new block and committing it to the network). As for the process of completing a transaction, first the user has to go to their bitcoin app, such as bitcoin core, and complete registration. Once the user has their bitcoin account, they can request a transaction with someone else by creating a new bitcoin transaction and submitting the request. Transaction verification receives their transaction and verifies it, and if it is valid, it sends the transaction data to Blockchain via the P2P network transfer module network, which then broadcasts the transaction to all nodes under Mempool. After that, the miner gets the transaction data from the mempool, decodes and packages it into a new block and submits it to the network. The blockchain updates the block information, which means the transaction is successful, and the information about the successful transaction is sent back to the user via Bitcoin core. If the transaction fails to be validated at the beginning, the information about the failed validation is sent back to the user to try again or make another new transaction.

5.2 Use Case 2

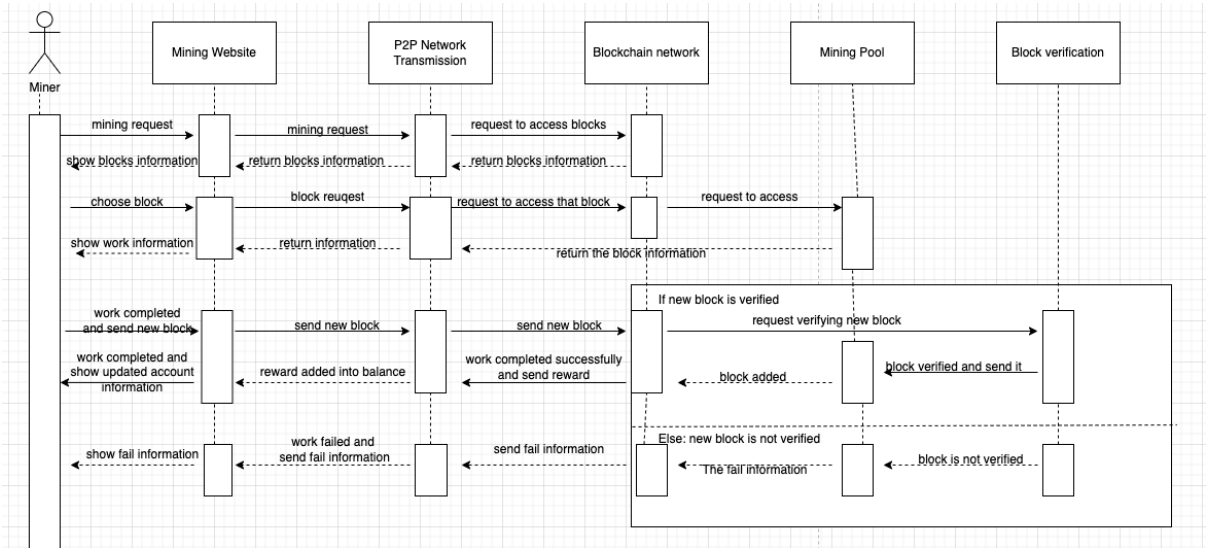

Figure 2. *Use Case 2 : Mining*

For concrete architecture of analysis of use case: Mining, there are six main components (other than the user), which are Mining Website (interface/app to interact with miners), Block verification (verifies the regularity, legitimacy and validity of new blocks), P2P Network Transmission

(responsible for communicating with the blockchain and transmitting data) , Blockcoin Network (stores and manages blockchain data and communicates with all nodes within the block), and Mining pool (manages the miner pool and collects proof of work). As for the process of completing a transaction, firstly miners have to register on the mining website and verify their miner identity. Once they have an account, they can send mining requests through the website, and the P2P network transmission sends their requests to the Blockchain network and requests access to blocks, after which the Blockchain network sends information about existing blocks back to miners through the P2P network.Next, the miner can make a mining selection, and then sending the selection request to the blockchain network, the request will be sent to the Mining pool again, which sends the specific information of the blocks back to the miner. Once the miner receives the block information, he can decode and pack the block. After the work is done, the miner will send the new block to the blockchain network, and then the block verification will verify the new block. If the verification is successful, the new block will be added into the blockchain network and rewards will be sended into miner's accounts. If the verification is unsuccessful, the failure message is sent to the miner, and then he can choose to try again or give up the mining.

6 Conclusion

Reflection analysis is a method that includes observing a software system's behaviour while it is in use to comprehend how it functions and spot any possible security vulnerabilities or performance problems. Developers and system administrators may use it to learn more about how their system works and make changes to improve its performance, dependability, and security. Reflection analysis can assist programmers and administrators in Bitcoin Core, a cryptocurrency software implementation, by gathering performance measurements, monitoring the system in real-time, and pinpointing potential problems. This is crucial in a financial system like Bitcoin because performance and security are essential elements that must always be maintained.Developers may better understand how Bitcoin Core works and spot any performance issues or security holes by continually monitoring the software while it is in use. The system may then be improved using this knowledge, such as by refining the code to increase speed or adding more security measures to fend off possible assaults. Reflection analysis can reveal important insights about the system's general behaviour in addition to pointing out problems. For instance, it can assist developers in comprehending how users interact with the system and see any emerging patterns or trends. The user experience may then be enhanced, and the system's performance can be optimized using this information. In conclusion, reflection analysis is an essential tool for understanding how Bitcoin Core behaves when in use and enhancing its speed, dependability, and security. Developers and administrators may acquire valuable insights into how the system functions and make changes that improve the user experience and make the financial system more secure by continually monitoring the system and gathering data.

A key component of Bitcoin Core is concurrency, essential to the system's ability to manage numerous transactions, blocks, and users simultaneously. Concurrency dramatically improves the network's performance and efficiency by enabling the system to handle several activities simultaneously. The blockchain is a distributed database that serves as the foundation for the

peer-to-peer network on which Bitcoin, a decentralized digital currency, runs. Blocks, each including a collection of verified transactions, comprise the blockchain. The Bitcoin network has to handle a considerable volume of transactions and blocks fast and effectively to maintain seamless operation. Concurrency becomes helpful in this situation. Bitcoin Core can efficiently handle the significant amount of requests the network receives by allowing numerous transactions, blocks, and users to be handled simultaneously. The system's scalability and robustness are enhanced by this concurrency, which also allows the task to be split across several nodes. Moreover, concurrency enables Bitcoin Core to benefit from cutting-edge computer hardware, such as multi-core processors, which can run several threads concurrently. The system's overall performance is much enhanced by this parallel job execution, which enables it to handle more transactions per second. In conclusion, concurrency is an essential part of Bitcoin Core that allows the network to handle many transactions, blocks, and users at once. Concurrency improves the network's overall performance and efficacy by utilizing contemporary computer hardware and distributed processing, assuring the efficient operation of the Bitcoin network.

In conclusion of bitcoin wallet analysis, the concrete architecture of Bitcoin wallets is complex and multifaceted, requiring careful consideration of the components and security measures that make up these software programs. The core wallet functionality is responsible for private key management, transaction signing, and communication with the Bitcoin network, while the user interface is responsible for providing an easy-to-use interface for users to interact with their funds. Security measures like multi-factor authentication, encryption of private keys, and hardware wallets are essential for protecting users' funds against a wide range of security threats. By understanding the architecture and security measures of Bitcoin wallets, users can make informed decisions about which wallet to use and how to protect their funds.

In conclusion, for concrete architecture analysis of bitcoin core use case 1 - Making transactions, the components of bitcoin core are the user, Bitcoin core, transaction verification, P2P network transmission, blockchain network, and miner. The process of completing a transaction involves creating a new transaction request, verification, P2P network transfer, broadcasting the transaction to all nodes, mining, and updating the blockchain. If the transaction is successful, the user receives information about the successful transaction via Bitcoin core. If it fails, the user is notified to try again or create a new transaction. As for the concrete architecture analysis of the use case "mining" for Blockchain. The components are the mining website, block verification, P2P network transmission, blockchain network, and mining pool. The process of completing a mining request involves verifying identity, sending requests to the blockchain network, making a mining selection, decoding and packing the block, and submitting the new block to the blockchain network for verification. If the verification is successful, the new block is added and the miner receives rewards. If it fails, the miner can try again or give up.

7 References

1. Satoshi Nakamoto.(n.d). Bitcoin: A Peer-to-Peer Electronic Cash System. Retrieved from https://bitcoin.org/bitcoin.pdf
2. Transactions — Bitcoin. (n.d.). Retrieved from https://developer.bitcoin.org/reference/transactions.html
3. P2P Network - Bitcoin. (n.d.). Retrieved from https://developer.bitcoin.org/reference/p2p_networking.html
4. Coinbase. (n.d.). What is a Bitcoin wallet? Coinbase Help. Retrieved March 24, 2023, from https://help.coinbase.com/en/coinbase/getting-started/crypto-education/what-is-a-bitcoin-wallet
5. Antonopoulos, A. M. (2014). Mastering Bitcoin: Unlocking Digital Cryptocurrencies. O'Reilly Media, Inc. Retrieved from https://www.oreilly.com/library/view/mastering-bitcoin/9781491902639/ch04.html