CISC 332/326 - Assignment 3 - Report

Bitcoin Core – Architectural Enhancement

Wednesday, April 12th, 2023

Deluca, Seb – 20sad4@queensu.ca

Huang, Robbie – 20rh1@queensu.ca

Kim, Ethan – 20eik@queensu.ca

Leyne, Aidan – 19afl@queensu.ca

Liang, Sean – sean.liang@queensu.ca

Lomonaco, Artie – 19al63@queensu.ca

Contents

Abstract

This report focuses on our proposed enhancement of Bitcoin Core's architecture to better improve efficiency of the system as a whole: The combination of the P2P subsystem and the Consensus Validation subsystem. After discussing the proposed enhancement, we suggest two possible ways to implement the enhancement within Bitcoin Core's architecture. Following this, we perform a SAAM analysis wherein we assess the impact of each variation of the implementation on various stakeholders and non-functional requirements. Once we have settled on a particular implementation, we then discuss the potential effects that the implementation of the enhancement will have on the overall system, and present two use cases. Finally, we discuss the possible risks that come with the implementation of the enhancement, and also briefly discuss the lessons learned.

Introduction and Overview

This report is aimed at examining the current Bitcoin Core system, finding shortcoming in the system and introducing a constructive enhancement for the platform. After reviewing the concrete architecture, we found a pair of subsystems serving similar purposes and possessing similar interaction with other subsystems. This prompted the idea for the assimilation of the two subsystems with the goal of streamlining the platform, code base and possibly introducing some performance benefits.

The Bitcoin Core system leverages the P2P architectural style. P2P is shorthand for Peer-to-Peer, a decentralized platform wherein individuals utilizing the service interact directly with each other, rather than having to interact with each other through a server. Bitcoin Core also utilizes the Publish/Subscribe architectural style for handling transactions, among other things. Our concrete architecture is derived accordingly, and the enhancement maintains this architecture.

For this enhancement we will propose two methods of implementation and perform a SAAM analysis to determine the better route. This analysis will outline the reasons one method of implementation remains stronger than the other while carrying into our analysis for affected subsystems and files, and how the enhancement will alter the way the current subsystems and calls interact. We will also be looking at some of the new risks brought-on by the enhancement and proposing new methods to test the enhancement in order to ensure that the reliability of the Bitcoin Core system remain largely uncompromised to the user.

While the enhancement, methodology and plan of action may seem unnecessary and solely potential for new risks, we do maintain the position that it is an enhancement for the Bitcoin Core platform and will improve efficiency, performance and the overall code-base.

Proposed Enhancement

The enhancement to the Bitcoin Core platform proposed in the following document consists of the combination of the Peer-to-peer and consensus validation subsystems. This enhancement can be brought to the platform in two ways:

Creation of a new joint-subsystem: by creating an entirely new subsystem and eliminating both he P2P and consensus validation subsystems, we can present certain optimizations in the code that would not otherwise be possible. We can also add other integrated features, make use of advancements in technology and streamline other aspects for optimization.

Removing one system and expanding the other: in this case, we will be reutilizing code already developed, implemented and tested from the original release of Bitcoin Core. Any features that are not present will be carried over between subsystems but we believe this task should be minimal. The code base maintains most of its robustness and builds on an already existing code base.

Motivation for Enhancement

Our target goal with this enhancement is to improve the Bitcoin Core's network performance, scalability, security, and maintainability. Currently, P2P and Consensus Validation operate as two separate and distinct systems, which we feel is a point of inefficiency that can be improved upon. For example, the Consensus Validation subsystem requires nodes to download and process the entire blockchain, which can be slow and resource-intensive, and can lead to centralization since smaller nodes can't afford the required storage and computational power. Additionally, the P2P networking subsystem, which is responsible for propagating new blocks and transactions, can be prone to network congestion. This often leads to delays and reduced throughput. Integrating these subsystems will allow for more efficient transaction processing, reduce the risk of attacks on the network and transaction fraud, and simplify the codebase, making it easier to maintain. These improvements will help ensure the long-term viability of the Bitcoin network, making it a more robust and trustworthy blockchain platform.

SAAM Analysis

In this section a SAAM analysis on bitcoin core will be provided. In this analysis, the major stakeholders will be addressed, the important NFRs that concern each stakeholder, evaluate the two suggested ways to realize the enhancement, how they impact each NFR/stakeholder, and finally determine which enhancement will be the overall best in this analysis.

Stakeholder	**Important NFRs Regarding the Enhancement**
User	Safety: The new enhancement maintains the safety that bitcoin core already has integrated in its P2P network. Performance: The application and the network work as smoothly as possible. Usability: Bitcoin core should inform the user of changes, and not impede its usability.
Developers	Testability: The module should be programmed in such a way that it can be tested in a safe environment and/or virtually. The programmers are also looking for a testing environment that is easy to maintain and modify. Maintainability: The module should be easy to maintain, especially when new modules are added that the guardian needs to check.
Investors	Safety: This should ensure that bitcoin investors assets are safe, and will increase in value. Quality: The new approach is improving bitcoin core's systems and investors will keep on funding.

Attributes	**First Approach:** **Combine P2P and Consensus**	**Second Approach:** **Create a new subsystem**
Performance	High: this approach guarantees that the performance will still be high since this new bigger subsystem already has its previous performances	High: The performance of this new subsystem can improve upon the issues that stemmed from these two systems being separated and remove unnecessary dependencies and increase efficiency.
Safety This approach needs to identify errors,	High: The safety from the previous subsystems will be carried over. This is a high level of safety for user's transactions so it is high.	Low/Moderate: The safety for the transactions of this new system is low since some things may have been overlooked in development, on the other one the safety carries over.
Maintainability This enhancement should be easily maintained by developers	High: The maintainability is high since there is a lot of documentation of these two subsystems that we are merging. All the resources to maintain will not need to be overhauled.	Moderate: The maintainability for this completely redesigned subsystem may be slightly more difficult than previous. Since it has new functions and dependencies than the first approach it will be reasonably difficult to maintain at first.
Quality This approach must work properly as bitcoin is advertised to investors and users.	High: The quality for combining the two subsystems will definitely be high since we will be merging the two subsystems that have been working perfectly for the system.	High: The quality of this new subsystem may vary. Some functions may encounter bugs or issues upon deployment. This is a lesser quality since the previous subsystem that is combined, the components have been endlessly maintained and tested unlike this one. The quality may still be high since it is removing many issues of the subsystems.
Testability In this approach, every scenario should be easily testable.	Moderate: This approach is a large system since it is merging those two, but the previous testing code could be used for this approach, making the testability as good as it originally was. Maybe some new tests have to be created.	High: This approach has to test large systems of bitcoin core or at least adjust the testing since they will be merged. More resources will be needed since this will be a pretty large system, so it will be more difficult to test.

Based on this Analysis, we have decided to implement the first approach, combining the P2P with the consensus. The reason is since merging these two subsystems will save a lot of time versus creating a completely new subsystem from scratch that performs similar functions to the latter. Although the efficiency of the brand new subsystem could be much greater, since we will be combining the functions of P2P and consensus, eliminating many unnecessary dependencies, combining the two subsystems will be more beneficial since a lot of time will be saved. By that I mean development time. Combining the two subsystems will guarantee that the functions of both subsystems will still be relatively the same, they are just getting merged. All the tools for testing, maintenance, and quality check will already be very useful and will not need to be created from scratch. Some of these may need to be slightly tweaked and updated but since the subsystems are already made of two already used components it will not need to be that much.

Effects of the Enhancement

Performance

An important factor of Bitcoin is its performance in a multitude of aspects. For this proposal, we aim to improve its performance regarding verification by combining the P2P networking and consensus validation subsystems. Doing this will reduce the number of internal components that are being communicated with during the validation process, allowing for significant improvements in transaction verification speed, which would especially be apparent when scaled to a large number of users. This will reduce the bandwidth requirements for relaying data, leading to faster propagation times and reduced network congestion. Additional improvements to performance that are provided by our proposed enhancement are touched upon in the next section.

Scalability

Scalability is a critical concern for blockchain networks, as it determines how many transactions the network can handle at a given time. Bitcoin's current block size limit of 1 MB is a significant limiting factor for its scalability, as it imposes a hard limit on the number of transactions that can be included in each block. However, by combining the P2P networking and consensus validation subsystems, it's possible to improve the software's ability to handle a larger number of transactions and blocks. By improving the way transactions are propagated through the network, the time it takes for a transaction to be seen by all nodes can be reduced. Reducing the overhead of validating each block will allow nodes to process blocks at a higher rate. These, in turn, will allow for a higher throughput of transactions, allowing the network to support more users and transactions.

Security

Integrating the P2P networking and consensus validation subsystems together will greatly increase the overall security of the system. New blocks are quickly and reliably propagated to other nodes and since the amount of transmissions between the components will be reduced during the validation process, as well as the size of data being communicated, it will be a lot more difficult for attackers to intercept and interfere with the network. For example, “double-spending” attacks occur when a user attempts to spend the same Bitcoin more than once by broadcasting conflicting transactions to different nodes on the network. By validating transactions at the networking layer, nodes will be able to quickly detect and reject invalid transactions, reducing the risk of these attacks.

Maintainability

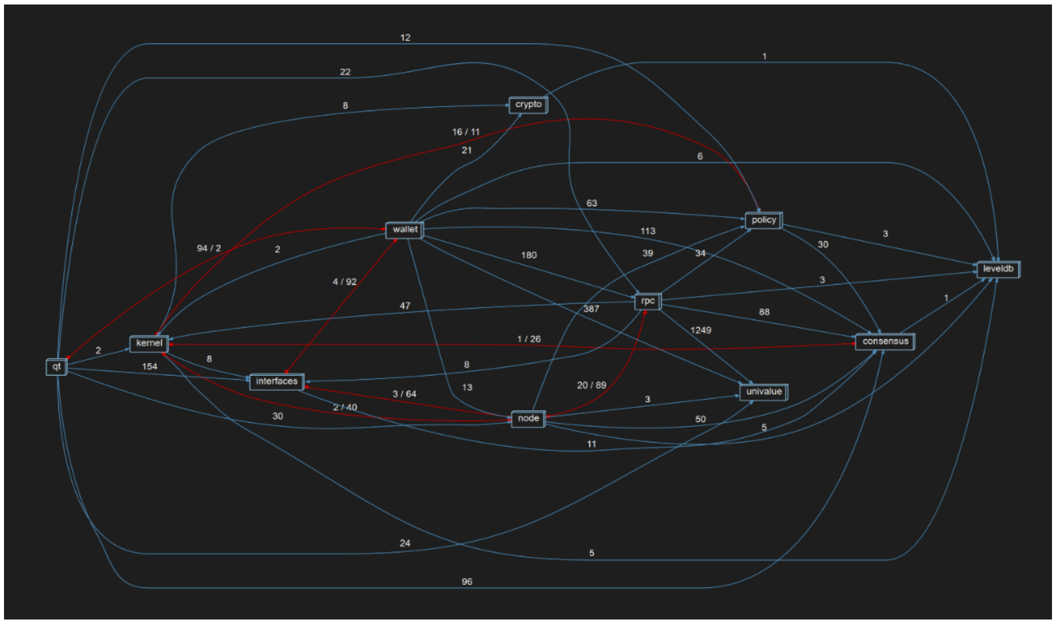
By eliminating an entire subsystem, the codebase of Bitcoin Core will be simplified, making it easier to maintain, update, and document over time. The system will become more modular, better allowing developers to easily add, remove, or modify functionality without affecting other parts of the system. An additional benefit of having a simpler codebase is that it could reduce the time and resources needed for testing and debugging, allowing developers to focus on other areas of the software, and improve Bitcoin Core as a whole.

Impacted Directories and Files

Without implementation, it is impossible to view the total extended of implemented directories and files at this time. Using our concrete architecture shown below, we can view some of the folders that will have direct impacts and others that are assumed to be impacted by Calls and Called-By.

Consensus: as this is one of the subsystems to be amalgamated, it is natural that this will see some of the largest changes. Depending on implementation, this folder could cease to exist or become much more central to the architecture. *Policy*: this folder takes the same shape as described above. It is central to the proposed enhancement and will see the most changes on a folder-level.

Wallet, leveldb and kernel: While we do not see the addition/subtraction of any files in these folders, there will be lots of changes to the calls through a refactoring process. For example, *kernel* shares direct calls with *consensus,* which at a minimum a refactoring will be necessary. Another example, *leveldb*, responsible for storage, will be need to be reconfigured for the newer, more efficient, structure of the platform.



Concurrency, External Interfaces and System Evolution

As the Bitcoin Core platform operates on a pub-sub model in relation to the blockchain, each peer is able to opt-in and out of the chain at any given time and return later. Internally, as we will see later in the report, it is paramount that all functionality remains concurrent with each other. This is not only for the blockchain requirements for transaction validation but also so that users are presented with accurate data, for example the state of their wallets.

The Bitcoin Core platform has two main external interfaces, the Bitcoin blockchain and the user GUI. The main functionality of the platform allows user to transact on the network, mine new coins through transaction validation and view its state. All this information is then presented to the user, the other external interface, so that they can utilize the platform.

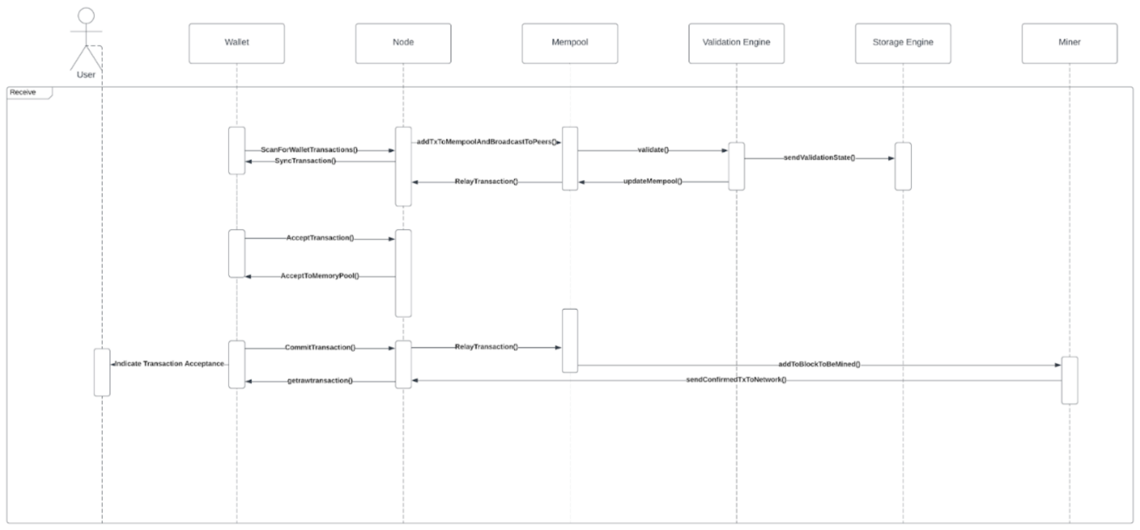
First released in 2014, the Bitcoin Core platform is presented by the Bitcoin Organization and developed through an open-sourced system and reducing transaction costs ten-fold. Developers from around the world collaborate on the public repository found on GitHub in order to present worldwide releases. One of the largest of these improvements took place in 2016 with the release of the *CheckSequenceVerify* fork.

Use Cases

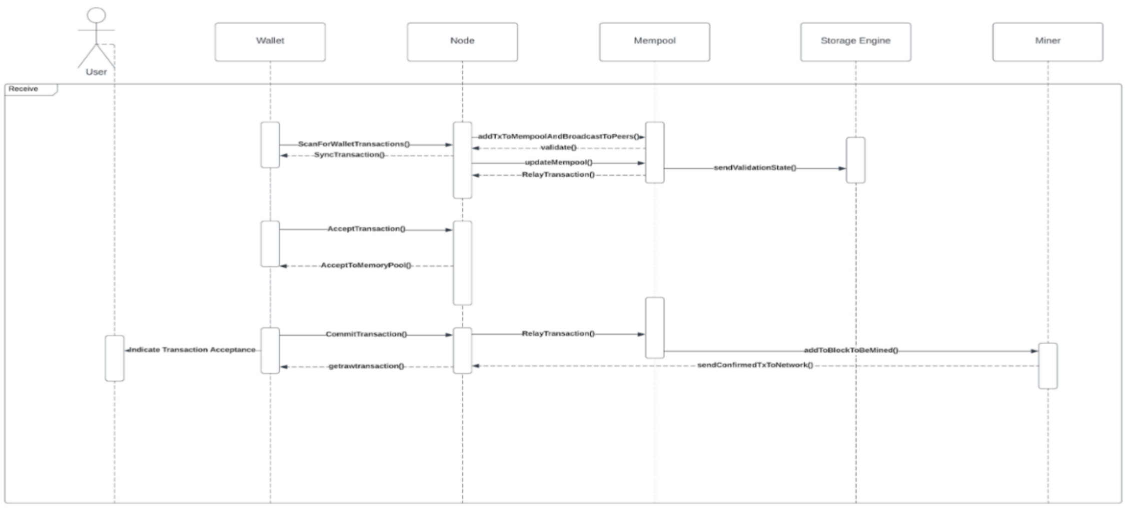
Use Case 1: Sending Money with Wallet

The below sequence diagram depicts the sequence of events that occur when a user is receiving money to their Bitcoin Wallet. It differs from our version of this diagram in our previous report because we needed to provide more detail in order to effectively convey the impact of our proposed enhancement. To provide said detail, we have expanded the previous 'Bitcoin Core' object into multiple sub-objects: Node, Mempool, Validation Engine, Storage Engine, and Miner.

Here is what such a sequence diagram would look like before our proposed enhancement.



and this is what the architecture will look like if the Validation Engine is merged into the P2P Network– which, in this diagram, is represented by the Node Object.



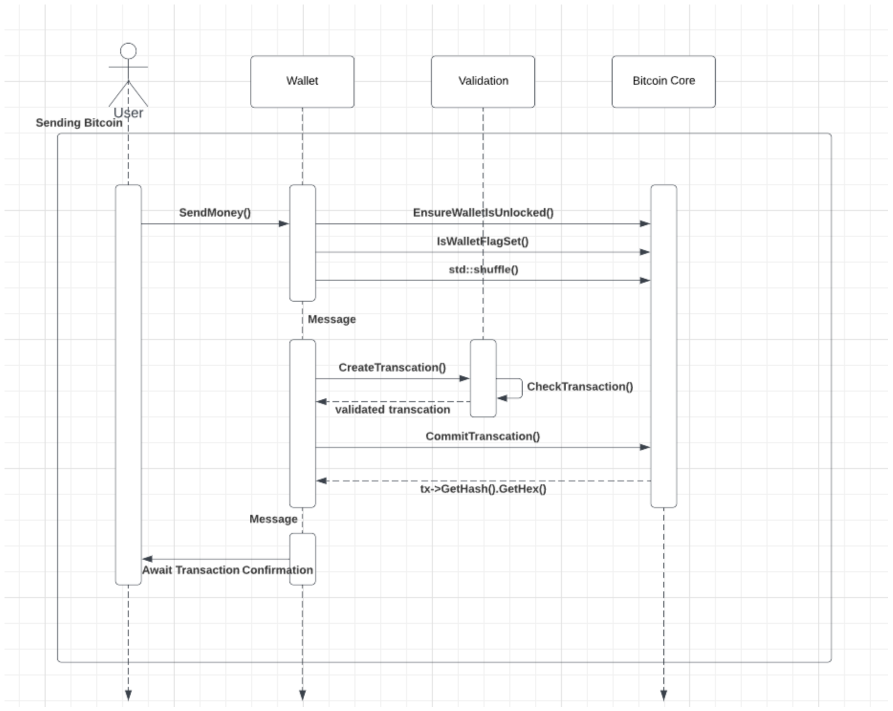
Because of the combination of the P2P Network and the validation engine, all function calls to and from the Validation Engine are now to and from the Node object.

Walkthrough

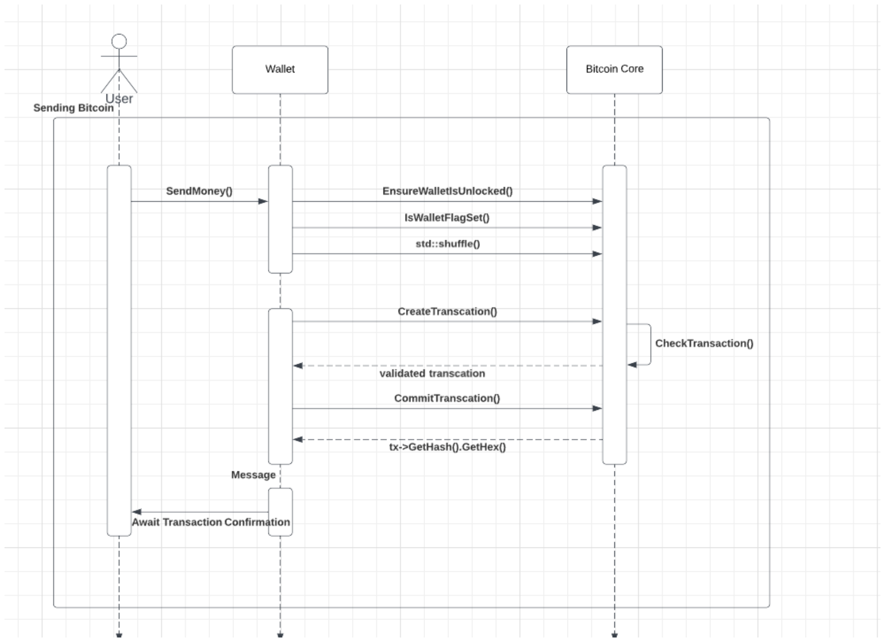
1. The wallet will query the Bitcoin Network for transactions using ScanForWalletTransaction to find a transaction to be accepted.
2. Once found, it will use the addTxToMempoolAndBroadcastToPeers function to add the transaction to the Mempool and begin the broadcasting of the new transaction to the other nodes on the network.
3. Then, the Mempool will validate the transaction with the validate function.
4. Once validated, the Mempool will be updated by the P2P Network (a part of Node) using the updateMempool function.
5. After this, we notify the Storage Engine that the new transaction is valid with the sendValidationState function.
6. Then, we broadcast the new transaction to the entire network with the RelayTransaction function.
7. Now, the wallet will accept the transaction with the AcceptTransaction function.
8. Once accepted, Bitcoin Core will accept it into the Mempool using the AcceptToMemoryPool function.
9. The wallet then commits the transaction with the CommitTransaction function– which adds the newly created transaction to the wallet's list of transactions and updates its balance.
10. To do this, the Node will use the RelayTransaction function to relay the transaction to the Mempool.
11. Then, the Mempool will add the transaction to a block to be mined with the addToBlockToBeMined function, where it can be mined by Miners on the Network.
12. Then, once the block is mined, the Miner object will confirm the transaction with the sendConfirmedTxToNetwork function.
13. Now, the balance of the user's wallet will reflect the updated transaction.

Use Case 2: Receiving Money with Wallet

The sequence diagram here shows receiving bitcoin from a user from before we added our enhancement onto it. We can see that this is a bit different from our original as we provided more detail to better depict what is happening with validation. This meant the inclusion of the Validation Engine object, where after the transaction is created it will be checked to see if it is a real transaction. After it will allow for the transaction to be committed.



After the implementation of the enhancement where merge between P2P and validation is made the diagram will look like this:



Walkthrough

1. The user will click on the ‘send money’ button in the UI of the wallet
2. The wallet application, ‘Wallet’ will then communicate with Bitcoin core to ensure that the wallet is unlocked using the ‘EnsureWalletIsUnlicked()’ function.
3. The ‘Wallet’ will then run ‘IsWalletFlagSet()’ function to check if the specific flag set for the wallet to check if it has private keys enabled or not.
4. The ‘Wallet’ runs the ‘std::shuffle’ method to randomly shuffle the recipient list into a new order to ensure that the order that outputs for transactions are created is randomized. This is done in an effort to make it more difficult for an attacker to identify the transaction
5. The ‘Wallet’ then creates the transaction using the ‘CreateTransaction()’ function. The function will construct a transaction that will meet the requirements necessary.
6. Afterwards it will be sent to the bitcoin wallet where it will not need to be checked within the ‘Validation Engine’ instead once it is given to ‘Bitcoin Core’ it will automatically validate using ‘CheckTranscation()’ function.
7. Once it is validated using that function it will allow the ‘CommitTransaction()’ to be called which will update the ‘Wallet’ state to reflect the new transaction (updating balance)
8. Bitcoin core will run ‘tx->GetHash().GetHex()’ function which will do two things, ‘tx->GetHash()’ returns the has of the transaction object, the second one is ‘GetHex()’ converts the hash into hexadecimal string.
9. Finally the transaction will return return a JSON response confirming the transaction.

Potential Risks

As the proposed enhancements reside at the center and comprise what could be considered the nervous system of the Bitcoin Core platform, these changes do pose a certain level of risk. First, when merging subsystems, special attention must be paid to all calls in and out ensure that all prior dependencies are transferred and refactored accordingly. Failure to do so will result in total failure of the platform. This risk can be mitigated through the use of various testing methods such as whitebox, blackbox and flow-through user testing. Edge cases remain problematic in this case and will continue to pose a risk. Unfortunately, at this time, we do not see a method capable of ensuring complete transfer and seamless performance and it is assumed that bug will be found after rollout.

Secondly, performance should be considered when evaluating the risks of the proposed enhancement. While the combination of subsystems would in theory provide greater performance through new available optimizations in the code, this also presents a risk of bottlenecking as one service is processing far more activity than before. However, a larger allocation of resources to the combined subsystem should compensate in full for this risk.

Lastly, availability and redundancy do both raise items relating to risk. As the enhancement proposes a much more central processing unit for the platform, if it were to fail, the impact would be much wider-felt than before. Currently, with the exception of a few edge-cases, we see the same issues with the two dedicated subsystems, however, with the enhancement there is a much larger guarantee of issues if the combined subsystem were to fail.

Plans for Testing

In evaluating the integration, performance, and security of combining the P2P and consensus validation components, Testnet will be utilized. Testnet is a blockchain system that is identical to Bitcoin but without any real-world value. The evaluation will be conducted using regression testing at a small scale to mitigate the risk of exposure to large-scale mishaps.

Given the stochastic risk inherent in combining two components, security risks must be carefully considered. Thus, small-scale testing in Testnet will be utilized to eliminate any possibility of data breaches and theft. Through the creation of a network of pseudo-users, cyber-security professionals can simulate common blockchain network attacks such as sybil, eclipse attacks, and eavesdropping between nodes.

Furthermore, the integration of two components can lead to significant performance bottlenecks. Therefore, careful logging and analytics of transactions will be conducted to assess the impact of the update. The primary metrics to be analyzed are transaction throughput and network latency. Transaction throughput, defined by the number of transactions processed within a given timeframe, will be used to assess the overall network performance. The secondary metric, network latency, will be utilized to evaluate whether or not the integration of components provides a more optimized use of bandwidth.

Limitations and Lessons Learned

Finding an enhancement to propose has not been easy. With such a large and established platform, changing any element is a considerably daunting task. When beging to search for a possible enhancement, the group felt that the system has already had the opportunity to complete multiple product lifecycles; making it much harder to find elements where we see opportunity for improvement. Further, we had to make sure that improvement within the platform maintained scope and reasonability. Changing the protocol itself violates reasonability as it would create an entirely new chain and coin while a complete overhaul of the system violates scope and it would be considered a new system. Finding a niche and executing a well constructed plan is difficult.

The importance of teamwork was also highlighted during our project. Managing our time effectively and aligning our schedules proved to be a challenge. The busy schedules of all team members necessitated the holding of asynchronous meetings to move the project forward. Moreover, communicating with the TA was challenging, requiring persistent email correspondence to organize team-wide convenient meetings. Our experience taught us that effective teamwork is vital in achieving project goals and delivering quality results.

Building accurate sequence diagrams highlighted the importance of the course material. In creating the sequence diagram, we had to employ strong deductive reasoning, based on the insights derived from the Understand program. We learned that effective utilization of the documentation is crucial to understanding the components and linking them in a communicable manner within the report. Our experience taught us that investing time and effort in studying the materials provided is crucial in creating a robust software architecture.

Conclusion

Overall, this report focused on our proposed enhancement of Bitcoin Core’s architecture to better improve efficiency of the system as a whole: The combination of the P2P subsystem and the Consensus Validation subsystem. After discussing the proposed enhancement, we suggested two possible ways to implement the enhancement within Bitcoin Core’s architecture. Following this, we perform a SAAM analysis wherein we assess the impact of each variation of the implementation on various stakeholders and non-functional requirements. From this, found a single solution and discussed the potential effects that the implementation of the enhancement will have on the overall system. Further, we presented two use cases where the feature is changed can be seen. Finally, we discussed the possible risks that come with the implementation of the enhancement, and the lessons learned.

Data Dictionary, Naming Conventions and Abbreviations

RPC: Remote Procedure Call is a protocol for requesting services located in another machine without requiring the details of the network that machine is located on.

P2P: Peer-to-Peer commonly refers to the decentralized ledger protocol used in blockchain technology; the name derives the connection of individual peers, lacking a central authority.

Subsystem: An individual computational environment where resources and workflow are coordinated.

Payment processors: Middlemen in the transaction between payee and merchants on the Bitcoin network.

Merchants: Business that sells products or services along the network.

Querying: Process of requesting information along the bitcoin network.

Consensus verification: Consensus verification is the process by which Bitcoin miners compete to solve a mathematical puzzle, ensuring the validity of new transactions added to the blockchain.

Decentralized: Decentralization is the process of distributing power and control away from a central authority, empowering a network of participants to collectively make decisions and govern the system.

Wallet: A stash for cryptocurrencies and crypto derived assets.

Block hash: The block hash is a unique identifier that is generated for each block in the Bitcoin blockchain, using a cryptographic algorithm to ensure the block's integrity and immutability.

Block height: Block height is the number of blocks that have been added to the Bitcoin blockchain before a particular block, indicating its position within the blockchain.

Fee rate: Fee rate is the amount of fees paid per unit of transaction size in a Bitcoin transaction, which determines the priority and confirmation time of the transaction in the blockchain

Pub/Sub: A pub/sub architecture is a messaging pattern where publishers send messages to topics or channels, and subscribers receive those messages, enabling asynchronous communication and decoupling of components in distributed systems.

Multisignature: Multisignature is a digital signature scheme that requires multiple signatures to authorize a transaction, providing enhanced security and accountability for digital applications such as cryptocurrency wallets.

References

Bitcoin Core integration/staging tree. (2023, February 20). GitHub. https://github.com/bitcoin/bitcoin/blob/master/doc/JSON-RPC-interface.md

Developer Guides — Bitcoin. (n.d.). Developer.bitcoin.org. https://developer.bitcoin.org/devguide/index.html

Transactions - Mastering Bitcoin [Book]. (n.d.). www.oreilly.com. Retrieved February 20, 2023, from https://www.oreilly.com/library/view/mastering-bitcoin/9781491902639/ch05.html#:~:text=The%20wallet%20calculates%20the%20user

A Blockchain Glossary for Beginners. (n.d.). ConsenSys. https://consensys.net/knowledge-base/a-blockchain-glossary-for-beginners/

Bitcoin Core - Bitcoin Wiki. (n.d.). En.bitcoin.it. Retrieved March 25, 2023, from https://en.bitcoin.it/wiki/Bitcoin_Core#Version_history

Bitcoin Core integration/staging tree. (2023, March 25). GitHub. https://github.com/bitcoin/bitcoin/blob/master/doc/JSON-RPC-interface.md#:~:text=The%20RPC%20interface%20allows%20other

Kernel Crypto API Architecture — The Linux Kernel documentation. (n.d.). Www.kernel.org. Retrieved March 25, 2023, from https://www.kernel.org/doc/html/latest/crypto/architecture.html