MuesliSwap Open Transaction Chaining

MuesliSwapTeam

November 24, 2023

1 Introduction

The Cardano blockchain has emerged as a significant player in the decentralized application (dApp) landscape, offering a blend of security, scalability, and sustainability. However, like many blockchain ecosystems, Cardano faces challenges in optimizing transaction processing and dApp performance. This document focuses on addressing these challenges through the development of Open Transaction Chaining tooling, aiming to streamline and accelerate transaction processing in Cardano dApps.

1.1 Importance of transaction chaining for efficiency

The primary objective of this document is to explore and propose a series of enhancements to the Cardano-node and Nami wallet. These enhancements are centered around the concept of transaction chaining - a method that could significantly reduce transaction times, thereby enhancing the overall user experience and efficiency of dApps on the Cardano blockchain. Transaction chaining involves consuming transactions that are not yet confirmed on-chain but only exist in the local mempool. The concept of transaction chaining has become more popular with the introduction of open tooling that allowed inspection of the local mempool of the Cardano-node [2] and is presented in more detail in [5].

Without transaction chaining, transaction consuming a unique state, such as a shared liquidity pool, must wait for the previous transaction to be confirmed and included in a block before they can be processed. This introduces a significant delay in the transaction processing pipeline, up to 20 seconds, which can be detrimental to the user experience. Transaction chaining aims to eliminate this delay by allowing transactions to be processed before preceding transaction are confirmed in blocks, thereby decreasing the latency of the system.

1.2 Objectives and Overview

To achieve this, we will first delve into the existing functionality and codebase of both the Cardano-node and the Nami wallet, particularly focusing on aspects related to transaction building and submission. This analysis is crucial for identifying the potential areas for integration of transaction chaining mechanisms. Additionally, we will consider other

relevant tools, such as Ogmios, to determine if they necessitate adjustments to align with the proposed transaction chaining feature.

The subsequent sections will outline the scope of the necessary changes, followed by a detailed, step-by-step development plan for implementing these changes. The goal is to provide a clear roadmap that not only enhances the current transaction processing capabilities but also sets a foundation for future advancements in the Cardano ecosystem.

2 Existing Functionality and Code Base Analysis

2.1 Cardano-node

2.1.1 Overview over Cardano-node

The Cardano-node is the core component of the Cardano blockchain. It is responsible for the validation of transactions and blocks and the propagation of these blocks to other nodes in the network. The Cardano-node is written in Haskell and is open-source. It is the reference implementation of the Cardano blockchain and is used by all other tools and wallets that interact with the Cardano blockchain.

In this project we will focus on the part of the Cardano-node that is user-facing and used to show the current state of the UTxO and build and submit transactions. The Cardano-cli is the command line interface to the Cardano-node and communicates with the node via a local socket to achieve the mentioned tasks. However, there are as of now no user-facing tools that access or make use of the mempool in the local node – even though every node has a mempool that stores locally submitted and peer-to-peer shared transactions.

2.1.2 Transaction presentation and building

The Cardano-cli provides a command to query the current UTxO of a given address or filtered by a transaction hash and id. This command lists all UTxOs that are currently available to be spent by the given address or that were an output of the specified transaction. For this list, the local Cardano-node is queried for the current UTxO set. The mempool is not considered for this query by default nor is there a user setting to enable it. This leads to delays when trying to confirm that a transaction has been submitted successfully, since it is not shown to the user even though it is residing in the mempool.

For transaction building the Cardano-cli provides a command that allows the user to specify the inputs and outputs of the transaction. Since a few versions, the CLI offers a convenient build command that automatically evaluates a transaction, computes the required fees and sends left-over change to a specified address. This build command queries the local Cardano-node for the current UTxO set and uses it to build the transaction and evaluate potential Smart Contracts that need to be executed. Similar to the listing of UTxOs, the mempool is not considered for this query by default nor is there a user setting to enable it. This leads to delays when trying to build a transaction that is based on a transaction that is still in the mempool.

Paradoxically, the node allows the user to submit transactions that consume UTxO still residing in the mempool. Therefore, the submit command of the Cardano-cli does not need to be changed to support transaction chaining. This also implies that we do not need to change the validity check of transactions in the node to support transaction chaining, and thus do not require any significant changes to the core ledger logic.

2.1.3 Overview over Nami Wallet

Nami Wallet stands out as a pioneering browser-based wallet extension, specifically designed as an open and intuitive interface to the Cardano blockchain ecosystem [1]. This non-custodial solution offers users a high degree of control and security over their assets. It allows users to store and manage Cardano Native Tokens (CNTs), delegate their stake and interact with decentralized applications (dApps) on the Cardano blockchain.

Nami Wallet has a history of defining the de-facto standard on Cardano, with the first implementation of the Cardano Improvement Proposal (CIP) 30, which introduced the concept of a unified wallet API for Cardano dApps [4]. Since its implementation is open-source, it serves as a good starting point for the development and standardization of transaction chaining features.

2.1.4 Transaction presentation and building

Currently, Nami wallet is a single address wallet. This means that all funds are kept in a single combination of public key and stake key. The history of transactions of the wallet is fetched from Blockfrost. This only considers transactions that are confirmed and included in a recent block and frequently causes delays. This presents an opportunity for adding transaction that are still in the mempool.

When building transactions, Nami wallet fetches all open UTxOs of the wallet from Blockfrost and uses them to build the transaction. This also does not consider whether (i) UTxOs have potentially been already spent in a pending transaction in the mempool and (ii) whether there are UTxOs available in the mempool that could be used to build the transaction. This presents the second opportunity for adding transaction chaining support.

Apart from this, the wallet also displays the current value of assets in the user wallet. This display is based on the output of the Blockfrost endpoints returning the current assets at a specific stake key. Since transactions in the mempool are not sure to be included in the blockchain we consider it not necessary to include them in the calculation of the current assets. Including the value and subsequently removing it would lead to a confusing user experience. Therefore, there is no need to include mempool updates for computed assets.

For the purposes of staking user funds, we also do not consider it necessary to include mempool transactions, since there is usually no urgency or need for quick updates regarding the stake pool. The stake pool delegation only takes effect several days after a submission.

2.2 Other Relevant Tools

2.2.1 Ogmios

Ogmios is a low-level tool that provides an alternative interface to the Cardano-node. It provides a mini-protocol for interaction with the mempool of the Cardano-node. Since ogmios is such a low-level tool and further provides its own mempool protocol, we do not consider it necessary to make any changes to it to support transaction chaining. Either way, the transaction building tooling in ogmios relies on the same code that the Cardano-CLI uses to build transactions. Therefore there is no need to seperately update ogmios.

2.2.2 Others

We did not find any other tools that would need to be updated to support transaction chaining. This is due to the fact that the Cardano-node is the reference implementation of the Cardano blockchain and therefore all other tools rely on it for their functionality. Therefore, most tools will automatically support transaction chaining once the Cardano-node supports it, or will be able to quickly adopt the changes to benefit from transaction chaining.

3 Scope of Changes

The changes conducted in the scope of this project concern the facilitation of transaction chaining in the Cardano-node and Nami wallet. The intended changes concern seamless integration of transactions that are still in the mempool into the transaction building process and the intuitive display of such transactions in transaction overviews as pending transactions. In both the Cardano-node (more specifically the Cardano-cli) and Nami wallet, display and transaction building are commonplace user-facing activities. We therefore consider these changes to be of high importance for the user experience of Cardano dApps.

The changes do not include changes related to implementations in the validity check of transactions, namely changes regarding the check whether such a chained transaction can correctly be applied to the current state of the blockchain. The reason is that from our current perspective, this task is not required. The cardano ledger implementation provided in the default Cardano-node already supports this functionality.

4 Development Plan for New Features

4.1 Implementation Plan for the Cardano-node

The list of UTxOs that is returned by the Cardano-cli is fetched from the local Cardano-node. This list is currently not including any UTxOs that are still in the mempool. The plan for this project is to introduce a new flag to the Cardano-cli that allows the user to

include mempool transactions in the list of UTxOs. This includes applying the mempool to the UTxO set that is returned by the Cardano-node, consuming UTxOs that are already spent in the mempool and adding UTxOs that are still in the mempool to the list of UTxOs. The required changes are expected to be minimal, since the Cardano-node already provides the functionality to apply the mempool to the UTxO set [1] and provides a protocol to fetch the current mempool from the node [2]. A small caveat is that there is no API to fetch the UTxO with the current mempool applied already. It will have to be determined whether it is more feasible to implement this API or to apply the mempool to the UTxO set in the Cardano-cli.

The transaction building command of the Cardano-cli is currently also not considering the mempool. For this command, the plan is to introduce a new flag that allows the user to include mempool transactions in the UTxO set that is used to build the transaction. This would be implemented by augmenting the UTxO set that is fetched from the Cardano-node for transaction building [3] with the mempool. The implementation would benefit from the same API in the Cardano-node that is used for the UTxO listing, for which the mempool is applied to the UTxO set.

4.2 Implementation Plan for Nami Wallet

The history of transactions of the wallet is fetched from Blockfrost.[4] This only considers transactions that are confirmed and included in a recent block and frequently causes delays. By using the recently developed endpoint of Blockfrost to provide access to the mempool of the Cardano-node[5], filtered by address, we can fetch all transactions that are still in the mempool and display them in the transaction history of the wallet as pending transactions.

When building transactions, Nami wallet fetches all open UTxOs of the wallet from Blockfrost and uses them to build the transaction.[6] This also does not consider whether (i) UTxOs have potentially been already spent in a pending transaction in the mempool and (ii) whether there are UTxOs available in the mempool that could be used to build the transaction. Again, by using the Blockfrost endpoint for mempool transactions, we can fetch all UTxOs that are still in the mempool and use them to build the transaction.

[1] https://github.com/input-output-hk/cardano-ledger/blob/afedb7d519761ccdd9c013444aa4b3e0bf0e68ef/eras/shelley/impl/src/Cardano/Ledger/Shelley/API/Mempool.hs#L109

[2] https://github.com/input-output-hk/ouroboros-network/pull/3404

[3] https://github.com/input-output-hk/cardano-cli/blob/ca5392ffffc33710155ecbd0fa4a63ccabdc9f99/cardano-cli/src/Cardano/CLI/EraBased/Run/Transaction.hs#L143

[4] https://github.com/berry-pool/nami/blob/fb05c0b1fba48188664409d3132ca730a5014bba/src/ui/app/components/historyViewer.jsx#L39

[5] https://docs.blockfrost.io/#tag/Cardano-Mempool/paths/~1mempool~1addresses~1%7Baddress%7D/get

[6] https://github.com/berry-pool/nami/blob/fb05c0b1fba48188664409d3132ca730a5014bba/src/ui/app/pages/send.jsx#L377

4.3 Integration Strategies

Effective integration strategies are paramount for the successful implementation of Open Transaction Chaining tooling in the Cardano ecosystem, specifically within the Cardano-node and Nami wallet. This section outlines the methodologies and approaches that will be employed to ensure that the new features are not only technically sound but also align well with the existing infrastructure and user expectations.

4.3.1 Interface Design

The goal is to design interfaces that allow seamless communication between the Cardano-node, Nami wallet, and other relevant tools. These interface must support the nuances of transaction chaining while maintaining backward compatibility with existing functionalities and being intuitive to use. This will enhance the user interface of the Nami wallet to intuitively accommodate new transaction chaining features, ensuring that users can easily access and utilize these enhancements without a steep learning curve.

4.3.2 Modular Implementation

We adopt a modular approach to implementation, ensuring that new changes can be integrated without disrupting the existing system operations. This approach allows for phased rollouts and easier troubleshooting.

4.3.3 Comprehensive Testing

We will conduct comprehensive testing to ensure that the new features are robust and function as intended. This ensures that the new features function correctly within the broader ecosystem and do not introduce unforeseen issues. The testing comprises a series of manual tests, using the wallet on testnet and mainnet environments and interacting with a variety of dApps.

4.3.4 User Feedback Loops

We establish feedback channels with end-users and developers to gather insights on usability and performance. This feedback is crucial for iterative improvements and ensuring the new features meet user needs.

By adhering to these integration strategies, we aim to create a robust and user-friendly environment for transaction chaining in the Cardano ecosystem. The ultimate goal is to enhance the efficiency and scalability of Cardano dApps, thereby contributing to the overall growth and success of the Cardano blockchain.

5 Impact on CIPs

The proposed changes in this document do imply some changes to the way that transactions are built and displayed. This has potential impact on standards that concern

interaction with external tools, i.e. for obtaining UTxOs or Transaction histories. Especially relevant for this context is CIP-30, which defines a standard for dApps to interact with wallets for transaction building and submission.

5.1 CIP-30

CIP-30 [4] defines a standard for dApps to fetch relevant information about wallets owned by the user. Moreover, it defines a standard for dApps to request the user to sign a transaction and submit it to the blockchain. We will outline below which changes or extensions may be necessary in the context of the changes proposed in Section 4.

We note that the biggest impact of the proposed changes is on endpoints that return specific UTxO sets or are related to transaction building and submission.

The first endpoint affected is the `getUTxOs` endpoint. We propose a boolean value to be added to the request that allows the user to specify whether mempool transactions should be included in the response. This value should default to false, to ensure that existing dApps are not affected by the change. However for dApps that want to make use of the new functionality, the value can be set to true. The reason that backwards compatibility should be preserved is that dApps may not be able to handle transactions that are not fully confirmed yet. An alternative would be to introduce a new endpoint that returns the UTxO set with mempool transactions applied. Either way, the dApp would need to make sure that the wallet supports the new feature before using it.

The second endpoint affected is the `getBalance` endpoint. Similar to the `getUTxOs` endpoint, we propose a boolean value to be added to the request that allows the user to specify whether mempool transactions should be included in the response. This value should default to false, to ensure that existing dApps are not affected by the change. In contrast to our decision not to include mempool transactions in the balance of Nami wallet, the decision whether they should be included for the dApp should be left to the dApp developer.

The third endpoint affected is the `signTx` endpoint. While the endpoint itself is not affected, the dApp should be aware that the transaction that is signed may not be valid yet. Whether this causes an issue depends on the wallet implementation. We propose adjusting the description of this endpoint such that dApp developers are aware of this potential issue. The same applies to the `submitTx` endpoint.

5.2 CIP-2

The changes proposed in this document do not affect CIP-2 [3]. While it may seem relevant for coin-selection, the algorithms for coin-selection are usually presented with a defined UTxO set. However we want to point out that the changes proposed in this document may affect the UTxO set that is presented to the coin-selection algorithm. Potentially, wallets may want to invoke coin selection algorithms with different UTxO sets, once with the UTxO set in the state of confirmed transactions and once with the UTxO set with an applied mempoolwith an applied mempool. Using UTxOs with an applied mempool can avoid submitting failed transactions due to insufficient funds or

re-spending UTxOs that are already spent in the mempool. At the same time, using mempool transactions is highly speculative and may result in an invalid transaction, should the mempool transaction be dropped from the mempool before the transaction is submitted.

5.3 Others

We did not find any other CIPs that would be affected by the changes proposed in this document.

6 Conclusion

In conclusion, the introduction of Open Transaction Chaining in the Cardano ecosystem, as implemented in the Cardano-node and Nami Wallet, represents a significant step forward in enhancing the efficiency and user experience of decentralized applications (dApps) on the Cardano blockchain. The proposed changes, focusing on integrating mempool transactions into both transaction building and display processes, promise to reduce transaction processing times and improve overall system responsiveness.

By enabling transactions to chain off of unconfirmed transactions in the mempool, we effectively minimize the latency associated with transaction processing. This feature is especially beneficial in scenarios where rapid transaction turnaround is crucial, such as in high-frequency trading or interactive gaming applications on the blockchain. Furthermore, the modular implementation and comprehensive testing strategies ensure that these enhancements are robust, reliable, and compatible with existing systems.

Looking ahead, the potential for further development in this domain is vast. Future enhancements could include more advanced mempool management techniques, further optimization of transaction processing algorithms, and even the integration of these concepts into other blockchain platforms beyond Cardano. The groundwork laid by this project opens up numerous avenues for research and development, potentially leading to more sophisticated blockchain infrastructures capable of handling an even wider array of applications with greater efficiency.

Overall, the implementation of Open Transaction Chaining in the Cardano ecosystem marks a pivotal advancement in blockchain technology. It not only addresses current challenges but also sets a precedent for future innovations in blockchain transaction processing, thereby contributing significantly to the evolution and maturation of blockchain technologies.

7 References

References

[1] URL: `https://namiwallet.io/`.

[2] cardanians.io. *Understanding the cardano mem-pool*. Nov. 2023. URL: https://cexplorer.io/article/understanding-the-cardano-mem-pool.

[3] Jonathan Knowles. *CIP 2 - Coin Selection Algorithms for Cardano*. URL: https://cips.cardano.org/cips/cip2/.

[4] Alessandro Konrad. *CIP 30 - Cardano dApp-Wallet Web Bridge*. URL: https://cips.cardano.org/cips/cip30/.

[5] Optim Labs. *Transaction chaining on cardano*. Sept. 2022. URL: https://optim-labs.medium.com/transaction-chaining-on-cardano-cdc38c1c73dc.