Scuola universitaria professionale della Svizzera italiana
Dipartimento tecnologie innovative

SUPSI

Sapphire Wallet: Blockchain Interoperability

Student

Nolli Manuele

Supervisor

Gremlich Giuliano

Co-supervisor

Guidi Roberto

Degree Course

Master of Science in Engineering: Computer Science

Year

2023 - 2024

Date

May 16, 2024

Contents

List of Figures

List of Tables

Abstract

Introduction

Chapter 1

Motivation and Context

Chapter 2

Problem

Blockchain technology lacks of *user experience*, every time a user wants to interact with a blockchain, he needs to sign a transaction, pay a fee, and wait for the transaction to be confirmed. This is a big barrier for the adoption of blockchain technology, compared to the traditional web 2.0 applications where the interaction is seamless, and the user does not need to know anything about the underlying technology.

The field is a very active research area, many solutions and chains have been proposed. In fact, millions of users and developers started to use them. [1] However, an important question arises when there are multiple chains: *which one should I use?*

Interoperability is the answer to this question, as it allows different chains to communicate with each other. In this way, the decision of which chain to use is not relevant and a business can benefit from the best features of each chain without losing market share. But how can this be achieved *without experienced users?*

The goal of this Master project is to analyse the current state of the art of blockchain interoperability and account abstraction, and to propose a solution that allows users to interact with multiple chains without the need to have any kind of knowledge about the used technology.

Chapter 3

State of the Art

This chapter presents the state of the art of blockchain interoperability and account abstraction, with an emphasis on EVM-compatible blockchains.[2]

In the first section, the concept of account abstraction is introduced, with a focus on the evolution of the proposals and an analysis of the Argent wallet to understand how account abstraction can improve the user experience.

The second section presents the state of the art of cross-chain communication, categorising the bridges according to the trust model, communication model, and asset transfer model.

Finally, the Internet Computer Protocol (ICP) is presented, as it is a promising solution for both account abstraction and cross-chain communication.

3.1 Account Abstraction

Different solutions have been proposed to improve the user experience (UX), including *Meta Transactions*, *Smart Contract Wallets*, and *Account Abstraction*. The results of the analysis done by the Institute of Information Systems and Networking at the University of Applied Sciences and Arts of Southern Switzerland (SUPSI) indicate that *Account Abstraction* presents the most promising solution for improving UX in decentralised systems, offering a comprehensive solution that enables smart contract logic to determine the fee payment and validation logic of transactions. [3]

In the context of Ethereum, there are two type of accounts: [4]

- *Externally Owned Accounts (EOA)*: controlled by anyone with the private key.
- *Contract Accounts*: A deployed smart contract. Controlled by the code.

EOAs are the only account type that can initiate transactions, while contract accounts can

only *react*[1] to transactions, which makes it difficult to do batches of transactions and requires users to always keep an ETH balance to cover gas. Other limitations include the lack of recovery options, the need to pay a fee for each transaction, and the risk of losing the private key. With these weakness, the user experience is not as seamless as in traditional web 2.0 applications. [5]

Account abstraction is a way to solve these problems by allowing users to flexibly include more security and better user experiences into their accounts. This can happen in three ways: [5]

- *Upgrading EOAs*: So that they can be controlled by Smart Contracts.
- *Upgrading Smart Contracts*: So that they can initiate transactions.
- *Separate the transaction system*: Adding a second and separate transaction system.

The first two solutions require an upgrade of the Ethereum protocol, while the third solution can be implemented without changing the protocol.[5] All possible paths will be discussed in the following sections.

Regardless of the route, the outcome is access to Ethereum via *Smart Contract Wallets*, either natively supported as part of the existing protocol or via an add-on transaction network. [5]

The use of *Smart Contract Wallets* unlocks new possibilities, including: [5]

- *Flexible security models*: Multi-signature, account freezing, transaction limits, allowlists, etc.
- *Recovery options*: Social recovery, guardians, etc.
- *Multi owner accounts*: Shared accounts, business accounts, etc.
- *Relayed transactions*: Sign a transaction and let someone else pay for it.
- *Batch/Multi transactions*: Execute multiple transactions in a single call.
- *Innovative User Experience*: Seamless interaction with the blockchain.

In the following sections, the most relevant Ethereum Improvement Proposals (EIPs) related to Account Abstraction are presented. In particular, it is possible to notice how the community is proposing different solutions to achieve the same goal.

[1] A smart contract can initiate a transaction by calling another smart contract, but the transaction is still initiated by an EOA.

3.1.1 EIPs

The community of Ethereum is actively working to improve the Blockchain. Everyone can propose an improvement through an EIP (Ethereum Improvement Proposal). An EIP is a design document providing information to the Ethereum community, or describing a new feature for Ethereum or its processes or environment. The EIP should provide a concise technical specification of the feature and a rationale for the feature. The EIP author is responsible for building consensus within the community and documenting dissenting opinions. [6]

In the context of Account Abstraction, the most relevant EIPs are in the category of *Standards Track EIPs*, which are changes that affect most or all Ethereum implementations. Standards Track EIPs can also be classified into three categories: [6]

- *Core*: Changes that require a consensus (hard) fork.
- *Networking*: Changes that affect network protocol, including changes that are specific to the network layer.
- *Interface*: Includes improvements around language-level standards, like method names and contract ABIs.
- *ERC*: Application-level standards and conventions, including contract standards such as token standards (ERC-20), name registries (ERC-137), URI schemes, library/package formats, and wallet formats.

3.1.1.1 EIP-86

Proposed by Vitalik Buterin[2] in 2017, EIP-86 is a *Core* proposal and can be considered the first step towards account abstraction. The goal of this proposal is to abstract the signature verification and the nonce scheme. This allows to develop smart contracts able to use any signature scheme in transaction verification. [7]

The default way to secure an Ethereum account is the ECDSA[3] signature scheme. However, EIP-86 allows to use any signature scheme, including multi-signature schemes and **custom cryptography**. [7]

It is important to notice that ECDSA cryptography is **not** quantum-resistant, the Peter Shor's algorithm can be used to break the ECDSA in polynomial time. [9]

The current status of EIP-86 is *Stagnant*, as it has been inactive for more than 6 months. Nevertheless, this proposal was an inspiration for the following EIPs.

[2] co-founder of Ethereum
[3] Elliptic Curve Digital Signature Algorithm [8]

3.1.1.2 EIP-2711

Proposed in 2020, The EIP-2771 is based on the EIP-2718 proposal, which introduces a new transaction type. The main features that EIP-2771 may introduce are: [10]

- *Sponsored Transactions*: Allow third parties to pay for a user's gas costs.
- *Batch Transactions*: Execute multiple transactions in a single call.
- *Expiring Transactions*: Set an expiration time that invalidates the transaction.

To achieve these features, the concept of **meta-transaction** has been introduced. The idea is that transactions signed by a user get sent to a *Forwarder* contract. The forwarder is a trusted **off-chain** entity that verifies that transactions are valid before sending them on to a gas relay. The forwarder passes the transaction on to a Recipient contract, paying the necessary gas to make the transaction executable on Ethereum. The transaction is executed if the Forwarder is known and trusted by the Recipient. This model makes it easy for developers to implement gasless transactions for users. [5]

The proposal is a *Core* EIP and it has been *Withdrawn* by the author. The reason for the withdrawal is that the EIP-3074, proposed with the help of the EIP-2711 author, is a more complete solution. [10]

3.1.1.3 EIP-2938

Proposed in 2020, The EIP-2938 is a *Core* proposal that introduces a new EVM opcode[4] called `PAYGAS`. As EIP-2711, also EIP-2938 is based on EIP-2718.

The main concept of EIP-2938 is to remove intermediaries during a relayed transaction. The implementations that need a Relayer are technically inefficient, due to the extra 21000 gas to pay for the relayer, economically inefficient, as relayers need to make a profit on top of the gas fees that they pay. Additionally, use of intermediary protocols means that these applications cannot simply rely on base Ethereum infrastructure and need to rely on extra protocols that have smaller userbases and higher risk of no longer being available at some future date. [11]

As it is possible to understand, the main concept of EIP-2938 is to **upgrade** what a **Smart Contract** can do.

Also the EIP-2938 is in a *Stagnant* status, as it has been inactive for more than 6 months. The community is currently favouring EIP-4337. [5]

[4] An Opcode is a machine-level instruction

3.1.1.4 EIP-3074

Proposed in 2020, EIP-3074 introduces two new opcodes to the Ethereum Virtual Machine (EVM): `AUTH` and `AUTHCALL`. The first sets a context variable authorized based on an ECDSA signature. The second sends a call as the authorized account. This essentially delegates control of the externally owned account (EOA) to a smart contract called *Invoker*. The main purpose of EIP-3074 is to **upgrade EOAs**. [12]

More specifically, `AUTH` enables the retrieval of a user's address from a signed message, effectively authenticating the user within EVM. `AUTHCALL`, on the other hand, simplifies the execution of authenticated calls by altering the sender of the transaction to the authorized address, thus simplifying the process of executing transactions that mimic smart contract logic directly from EOAs. [5]

This gives developers a flexible framework for developing novel transaction schemes for EOAs. A motivating use case of this EIP is that it allows any EOA to act like a smart contract wallet without deploying a contract (many EOAs may use the same *Invoker* contract). [12]

The *Invoker* contract is a trustless intermediary between the EOA and the rest of the Ethereum network. The contract can be used to implement a variety of features, such as *sponsored transactions*, *expiration*, *batch transactions*, etc. [12]

As stated in the EIP: [12]

> Choosing an invoker is similar to choosing a smart contract wallet implementation. It's important to choose one that has been thoroughly reviewed, tested, and accepted by the community as secure. We expect a few invoker designs to be utilized by most major transaction relay providers, with a few outliers that offer more novel mechanisms.

EIP-3074 is currently in *Review* status, and it is expected to be included in the next hard fork of Ethereum, named *Pectra*, expected in Q4 2024. [13]

3.1.1.5 ERC-4337

Proposed in 2021, ERC-4337 is an account abstraction proposal which completely avoids the need for consensus-layer protocol changes by creating **separate transaction system**. [14]

Instead of adding new protocol features and changing the bottom-layer transaction type, this proposal instead introduces a higher-layer pseudo-transaction object called a *UserOperation*. As illustrated by figure 3.1, users send *UserOperation* objects into a separate *mempool*. A special class of actor called bundlers package up a set of these objects into a transaction making a *handleOps* call to a special contract, and that transaction then gets included in a

block.

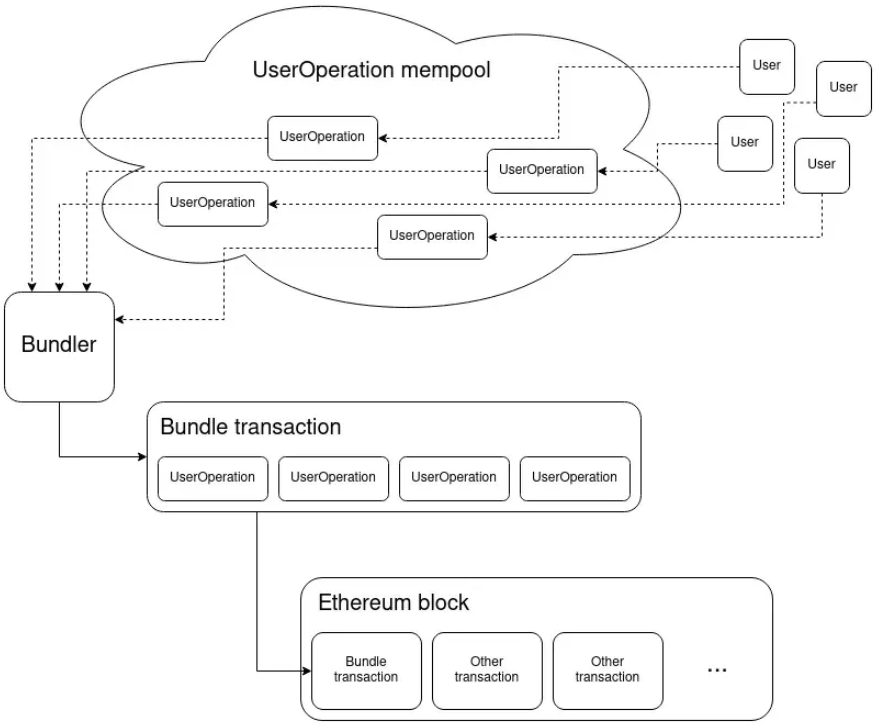

Figure 3.1: ERC-4337 Diagram

ERC-4337 also introduces a paymaster mechanism that can enable users to pay gas fees using ERC-20 tokens (e.g. USDC) instead of ETH or to allow a third party to sponsor their gas fees altogether, all in a decentralized fashion. [3]

The ERC-4337 is *Draft* status and the main start contract required was deployed in March 2023. [15]

3.1.1.6 EIP-7702

Proposed in 2024 co-authored by Vitalik Buterin, EIP-7702 is the latest proposal for account abstraction. The main goal of this proposal is to offer the same features as EIP-3074, but without the need of the two opcodes `AUTH` and `AUTHCALL`. The reason is that, as stated in the EIP, the two opcodes will not be used in the final solution for Account Abstraction where Smart Contract Wallets will eventually be implemented. [16]

EIP-7702 propose to a new transaction type that adds a `contract_code` field and signature, and converts the signing account (not necessarily the same that started the transaction) into a smart contract wallet for the duration of that transaction. [16]

This *Core* proposal is in an early stage, as it is in *Draft* status. But it is promising, as it is co-authored by Vitalik Buterin, one of the co-founders of Ethereum.

3.1.2 Argent

3.2 Cross-Chain Communication

3.2.1 Bridge classification

3.2.1.1 Trust model

3.2.1.2 Communication model

3.2.1.3 Asset transfer model

3.3 ICP: Internet Computer Protocol

Chapter 4

Problem Approach: Sapphire Wallet

4.1 Architecture

4.2 Blockchain

Smart contracts deployed on the blockchain.

4.2.1 Base Chain

4.2.2 Dest Chain(s)

4.3 Bridge

4.4 Backend

Microservices infrastructure.

4.4.1 API Gateway

4.4.2 Wallet Factory

WalletCreation.draw.io

4.4.3 Sapphire Relayer

4.4.3.1 Base Chain Relayer

TransactionExecution.draw.io

4.4.3.2 Dest Chain Relayer

All bridgeCall folder

4.4.4 Sapphire Portfolio

Problem: how to retrieve the wallet info on all the chains?

portfolio folder

4.5 Mobile Application: Sapphire Wallet

4.5.1 Wallet Creation

4.5.2 Wallet Recovery

4.5.3 Home Page

Write all the operations, balance of chains, button.

4.5.4 NFTs Page

4.5.5 Settings Page

guardian

Chapter 5

Results

Chapter 6

Conclusions

Bibliography

[3] L. Ambrosini, G. Gremlich, T. Agnola, L. Ronzani, R. Guidi, and T. Leidi, “Account abstraction: User experience in blockchain systems,” 2023.

[9] I. Stewart, D. Ilie, A. Zamyatin, S. Werner, M. Torshizi, and W. J. Knottenbelt, “Committing to quantum resistance: A slow defence for bitcoin against a fast quantum computing attack,” *Royal Society open science*, vol. 5, no. 6, p. 180 410, 2018.

Linkography

[1] “Essential blockchain statistics 2023.” (Dec. 2022), [Online]. Available: https://www.zippia.com/advice/blockchain-statistics/ (visited on May 6, 2024).

[2] “Ethereum virtual machine.” (), [Online]. Available: https://crypto.com/glossary/it/ethereum-virtual-machine-evm (visited on May 7, 2024).

[4] “Ethereum accounts.” (Oct. 2023), [Online]. Available: https://ethereum.org/en/developers/docs/accounts/ (visited on May 7, 2024).

[5] “Ethereum account abstraction.” (Mar. 2024), [Online]. Available: https://ethereum.org/en/roadmap/account-abstraction/ (visited on May 7, 2024).

[6] “Eip-1.” (Oct. 2015), [Online]. Available: https://eips.ethereum.org/EIPS/eip-1 (visited on May 14, 2024).

[7] “Eip-86.” (Feb. 2017), [Online]. Available: https://eips.ethereum.org/EIPS/eip-86 (visited on May 13, 2024).

[8] “Elliptic curve digital signature algorithm.” (May 2024), [Online]. Available: https://en.wikipedia.org/wiki/Elliptic_Curve_Digital_Signature_Algorithm (visited on May 13, 2024).

[10] “Eip-2711.” (Jun. 2020), [Online]. Available: https://eips.ethereum.org/EIPS/eip-2711 (visited on May 13, 2024).

[11] “Eip-2938.” (Sep. 2020), [Online]. Available: https://eips.ethereum.org/EIPS/eip-2938 (visited on May 14, 2024).

[12] “Eip-3074.” (Oct. 2020), [Online]. Available: https://eips.ethereum.org/EIPS/eip-3074 (visited on May 14, 2024).

[13] “Ethereum update: Pectra.” (Apr. 2024), [Online]. Available: https://github.com/ethereum/pm/issues/997#issuecomment-2045893316 (visited on May 14, 2024).

[14] “Eip-4337.” (Sep. 2021), [Online]. Available: https://eips.ethereum.org/EIPS/eip-4337 (visited on May 14, 2024).

[15] “Ethereum roadmap user experience.” (), [Online]. Available: https://ethereum.org/en/roadmap/user-experience/ (visited on May 14, 2024).

[16] “Eip-7702.” (May 2024), [Online]. Available: https://eips.ethereum.org/EIPS/eip-7702 (visited on May 16, 2024).