W3Smart Wallet

W3Smart Wallet Whitepaper

A revolutionary multi-chain wallet

Version 1.0
8/2023

Le Trong Dat datlt@smartosc.com

Table of Contents

Table of Figures

A. Introduction

1. Overview

W3Smart Wallet is an account abstraction wallet with a built-in marketplace, providing a comprehensive solution to gamers and game developers who want to venture into the blockchain gaming space. We offer a user-friendly crypto wallet, NFT game items marketplace, assets, and sales management space, and access to a user base of game players. Additionally, we make it easy for users to get on board with Web-3 gaming by providing an effortless experience.

2. Account Abstraction (AA)

Account abstraction presents a solution to enable users to incorporate enhanced security measures and user experiences into their accounts more flexibly. The term "account abstraction" refers to a proposal aimed at enhancing user engagement with Ethereum, and it's becoming a hot topic within the cryptocurrency community.

2.1. Types of Account Structures

Figure 1: Types of account structures in Ethereum

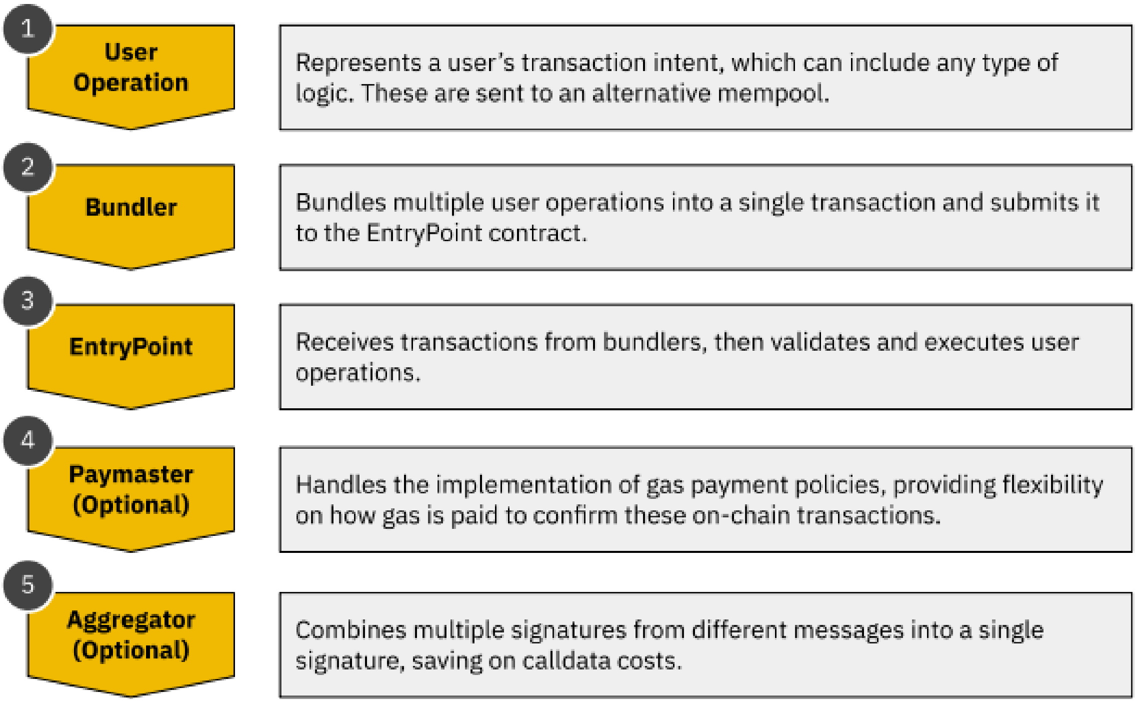
Externally Owned Account (EOA)	**Contract Account (CA)**
Signing accounts	Smart accounts
Controlled by private keys	Controlled by code
Unable to contain programmable logic	Contains programmable logic for execution

Hence, Account Abstraction aims to combine an Externally Owned Account (EOA) with a contract account (CA), allowing users to dynamically program their wallets using "**smart contract wallets**."

2.2. Technical Overview

To grasp the functioning of the procedure, the overall mechanism outlined in Figure 2 below elucidates the essential elements of AA. These attributes collaborate to enable developers to construct intelligent contract wallets that harmonize with on-chain decentralized applications (dApps).

Figure 2: Overview of AA



1 User Operation

Represents a user’s transaction intent, which can include any type of logic. These are sent to an alternative mempool.

2 Bundler

Bundles multiple user operations into a single transaction and submits it to the EntryPoint contract.

3 EntryPoint

Receives transactions from bundlers, then validates and executes user operations.

4 Paymaster (Optional)

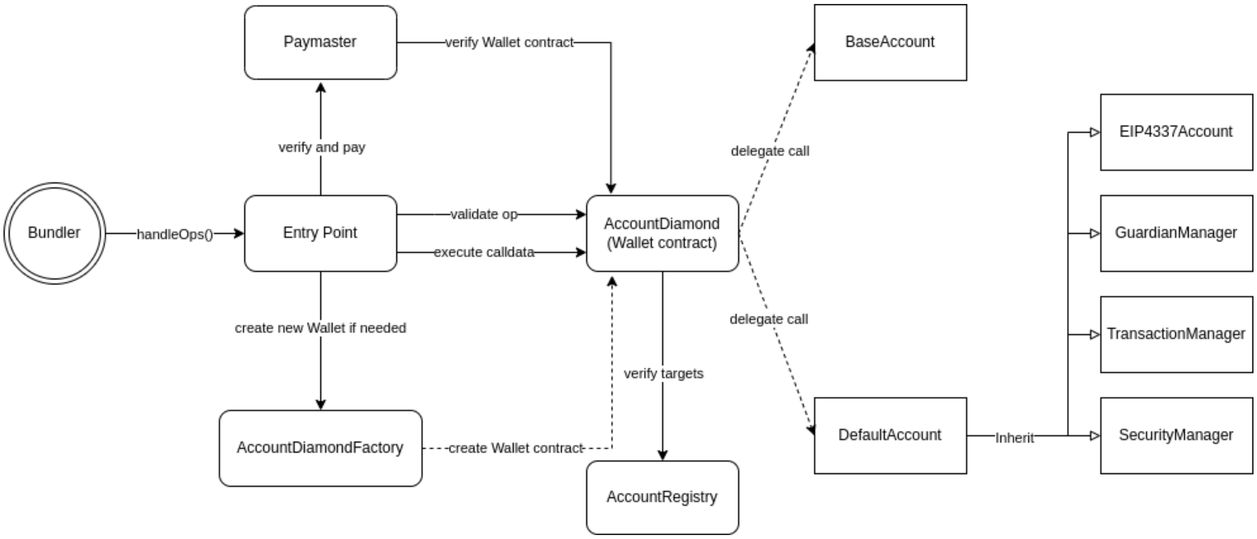
Handles the implementation of gas payment policies, providing flexibility on how gas is paid to confirm these on-chain transactions.

5 Aggregator (Optional)

Combines multiple signatures from different messages into a single signature, saving on calldata costs.

Smart Contract Architecture

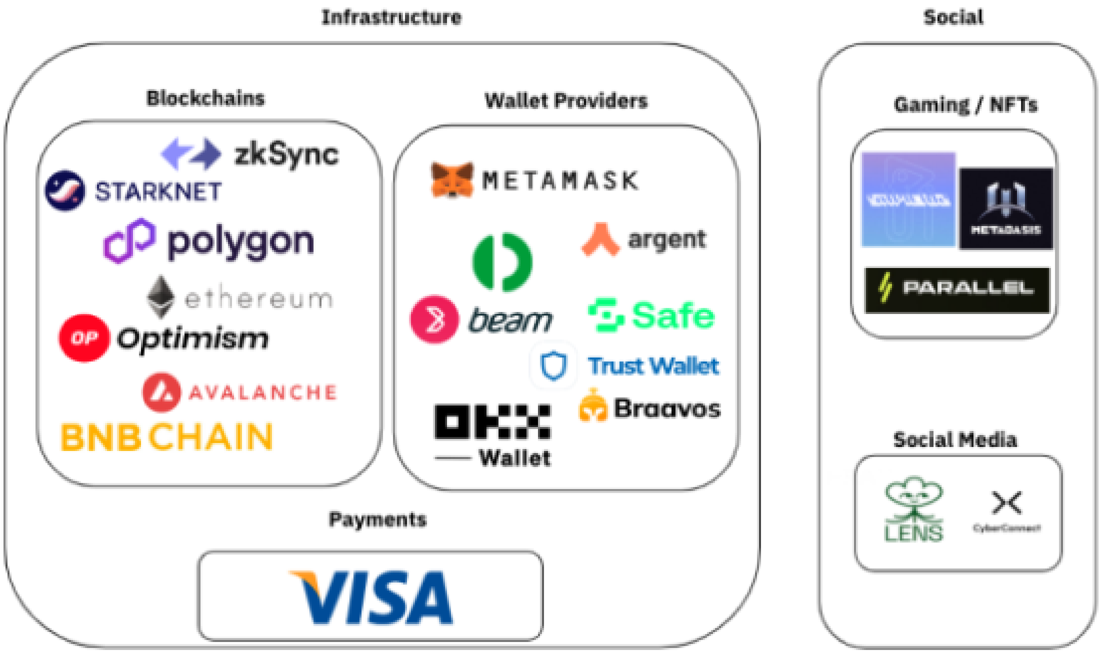
Figure 3: Smart Contracts architecture



2.3. The AA Ecosystem Overview

Over the last year, the AA ecosystem has witnessed swift growth, driven by developers acknowledging the demand for more streamlined user experiences and user-friendly engagements among wallets, apps, and users. Through the integration of AA, the objective is to refine the current framework and broaden opportunities for users to seamlessly participate in on-chain activities, promoting greater adoption in an instinctive manner. Progress in this domain can be categorized broadly into "infrastructure" and "social" advancements.

Figure 4: Current AA ecosystem



- **Blockchains**: EVM-compatible blockchains and scaling solutions such as zk-rollups that support AA.

- **Wallet Providers**: Wallets that leverage AA to provide user-specific functionalities.

- **Payments**: Provides fee abstraction capabilities through Paymaster contracts, where users do not need to own the blockchain's native tokens and can still execute digital transactions.

- **Gaming/NFTs**: Adopts AA with token standards such as trading with ERC-721, and ERC-1155 to improve the on-chain gaming experience and extend the utility of NFTs.

- **Social Media**: Incorporates forms of AA such as signature abstraction to enable users to continuously interact with the social network with their Web3 wallets.

3. Exploring Use Cases

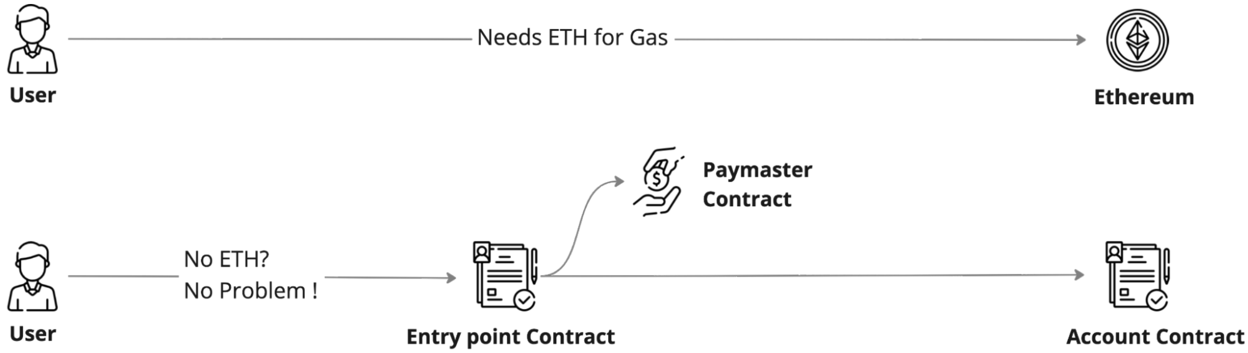
3.1. Paymaster

Having the native tokens of a blockchain is crucial for users who want to engage with a decentralized application. When sending a transaction on the Ethereum network, it is necessary to possess Ether to cover the costs of gas fees. Consequently, newcomers are required to buy these cryptocurrency tokens before they can commence using a dApp. This is a significant obstacle for bringing new users onboard.

How does Paymaster work?

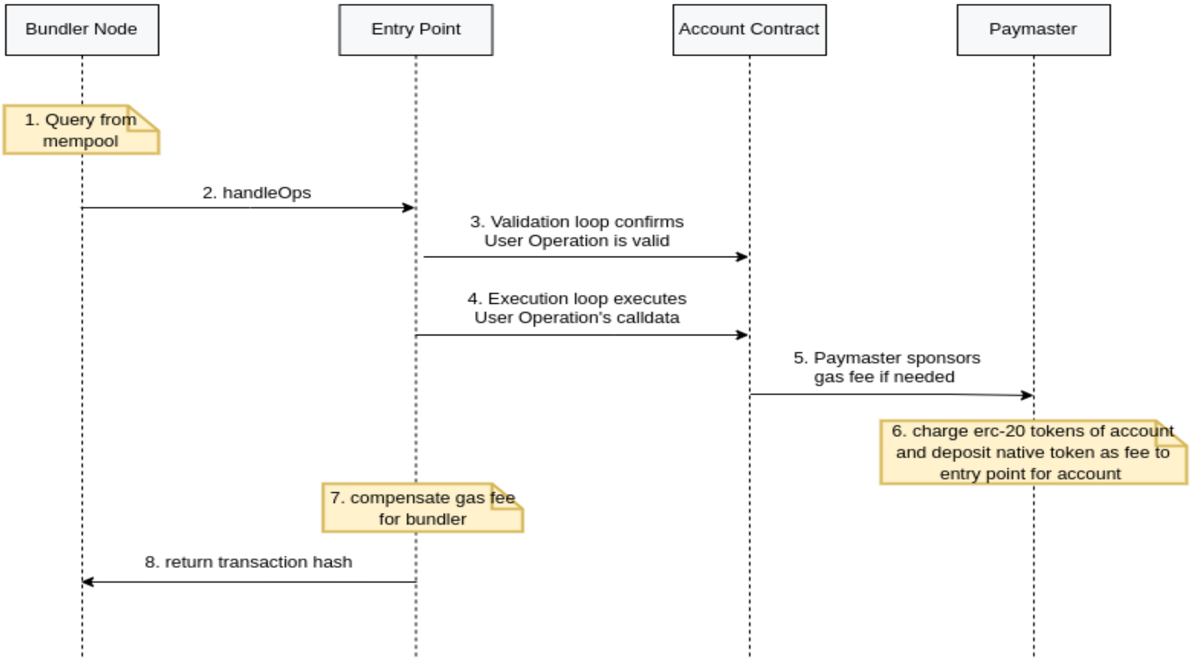
Paymaster is a special entity that can sponsor gas for an account if their conditions are met. This allows account holders to pay for transaction fees in many different ways without resorting to custodial solutions.

Figure 5: Paymaster high-level flow



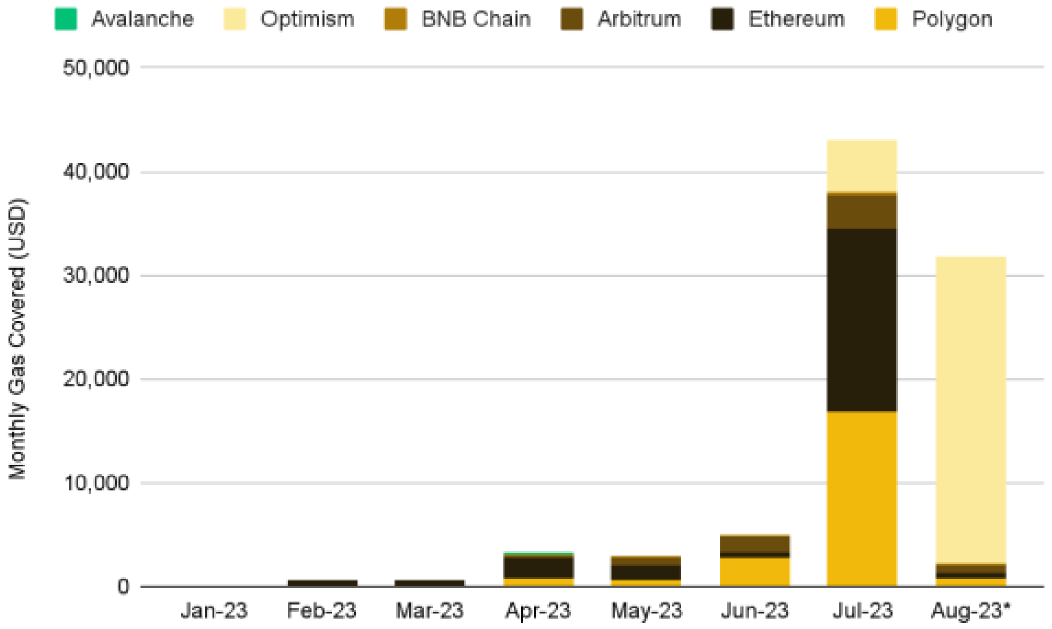
W3Smart Wallet incorporates a Paymaster-enabled gas sponsorship feature that enables users to execute transactions without having to own or know about native tokens

Figure 6: Paymaster contract flow


Bundler Node
Entry Point
Account Contract
Paymaster
1. Query from mempool
2. handleOps
3. Validation loop confirms User Operation is valid
4. Execution loop executes User Operation's calldata
5. Paymaster sponsors gas fee if needed
6. charge erc-20 tokens of account and deposit native token as fee to entry point for account
7. compensate gas fee for bundler
8. return transaction hash

Indeed, the utilization of Paymaster contracts within the on-chain environment has exhibited consistent expansion throughout the sector, notably led by **Optimism**. This trend has been further reinforced by recent initiatives like the **Beam wallet**, which was introduced on **Optimism** towards the end of July. In a parallel manner, the Beam wallet leverages **Paymasters**, facilitating users to cover gas fees using the currency employed in the transaction rather than the inherent token of the blockchain.

Figure 7: Monthly gas covered by Paymaster contracts has drastically risen this year



The volume of gas accommodated by these solutions has notably surged, indicating a rising desire for these intermediary services that streamline the user journey.

Consequently, by removing the intricacies tied to blockchain transactions, users are relieved from the exclusive requirement of holding the blockchain's native tokens just for gas fee payment. This enhancement enhances the current payment structure and establishes a more approachable and user-centric atmosphere for digital transactions.

Visa

Digital payments giant Visa experimented with Paymaster contracts to abstract away the basic blockchain interactions and improve the on-chain user payments experience through a self-custodial smart contract wallet. The proof of concept aimed to reduce friction for users to transact through their wallets and explore the “untapped potential” of digital transactions for consumers.

3.2. Wallet Management

Social Recovery

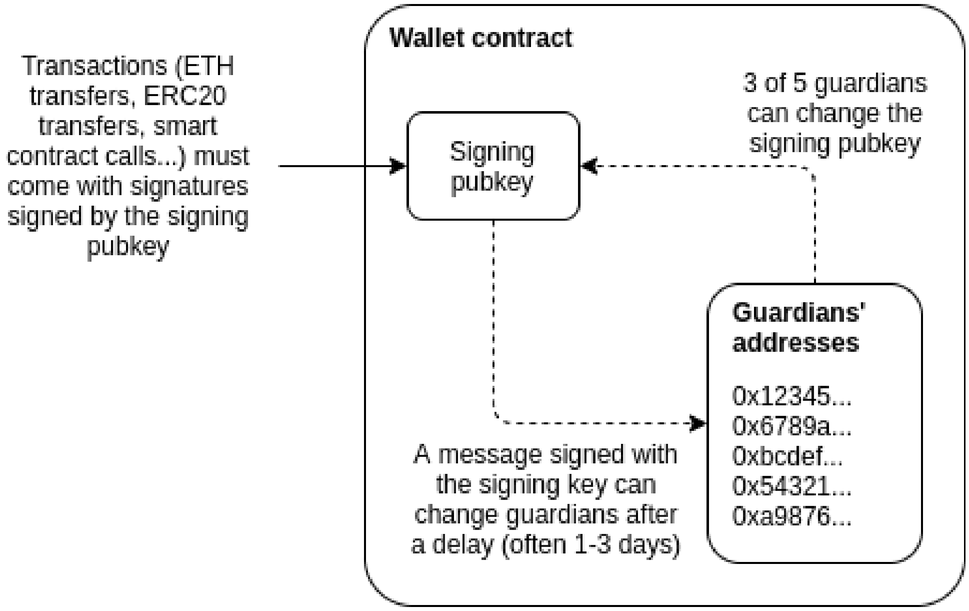
Social recovery is a way to recover your self-custodial wallet without using seed phrases. While with a custodial wallet, a third party keeps your private key, a self-custodial one allows you to have full control over your assets. This also means that once you lose your key (in the form of a seed phrase), you lose access to your assets.

The **W3Smart Wallet** security model is based on the power of guardians concept.

A guardian is an account (an EOA or a **W3Smart Walle**t account) that has been permitted by the wallet's owner to execute certain specific operations on their wallet:

- approve a guardian addition/revocation
- approve a whitelist's participant addition
- approve wallet unlocking
- approve a wallet recovery
- approve an untrusted transaction

Figure 8: How do guardians protect your wallet?


Transactions (ETH transfers, ERC20 transfers, smart contract calls...) must come with signatures signed by the signing pubkey
Wallet contract
Signing pubkey
3 of 5 guardians can change the signing pubkey
Guardians' addresses
0x12345...
0x6789a...
0xbcdef...
0x54321...
0xa9876...
A message signed with the signing key can change guardians after a delay (often 1-3 days)

Guardians **never have access** to the wallet's cryptocurrency.

Two-Factor Authentication (2FA)

Types of Guardian:

- **System guardian**: System guardian is an automated service provider that uses two-factor authentication (phone & email). Every wallet will come with this default guardian. It is added to the wallet when it is deployed onto the blockchain. Users can remove it at any time if they choose to keep their wallets secure with other guardians.

- **Other guardians:** Along with the system guardian, there are additional guardians (through the Adding a guardian process) that users choose to secure their wallets. Other guardians can be an EOA account or a smart account.

With **W3Smart Wallet**, our system acts as the first guardian to protect your wallet. When you initiate a multisig transaction, we activate 2FA by sending an OTP code to your email to verify your identity. Successfully verifying OTP code means the system guardian signs and approves your transaction request.

Freezing account

In case the wallet's owner suspects his account (i.e. device) is compromised (lost, stolen, …), he can use the Freezing Account feature to immediately lock the wallet to protect his assets.

Whitelists

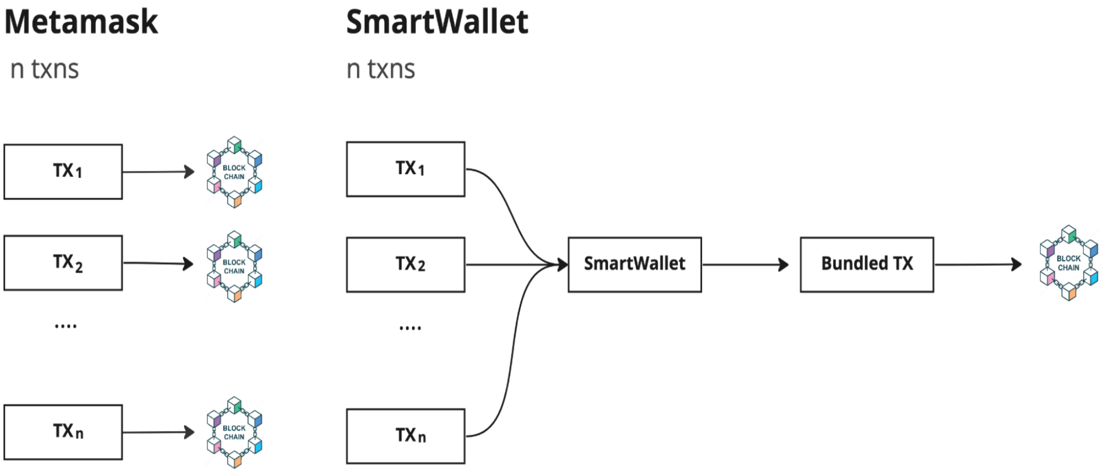
As the popularity of blockchain technology continues to grow, so does the demand for secure and reliable smart contract wallets. Smart contract wallets are designed to hold token assets securely on the blockchain and automate transactions based on predetermined rules. However, with the increase in popularity of smart contract wallets, there is also an increased risk of security breaches and fraudulent activities. To address this issue, **W3Smart Wallet** provides a whitelist feature.

A whitelist is a list of approved addresses or smart contracts that a user's wallet can send funds to or interact with while denying all others. The purpose of a whitelist is to provide an extra layer of security to prevent unauthorized transactions or interactions. Enabling a whitelist helps reduce the risk of malicious activities, thereby ensuring the security of users' assets.

If you are a user who frequently makes multiple transactions, you may have heard of batch transactions. We will explore the benefits of batch transactions and how they can help you save time and money.

Batch transactions

Figure 9: How does Batch Transaction work?



Batch transaction brings a lot of advantages for users:

- **Cost optimization**: Batch Transaction helps to reduce costs for users by reducing the number of transactions that need to be executed.

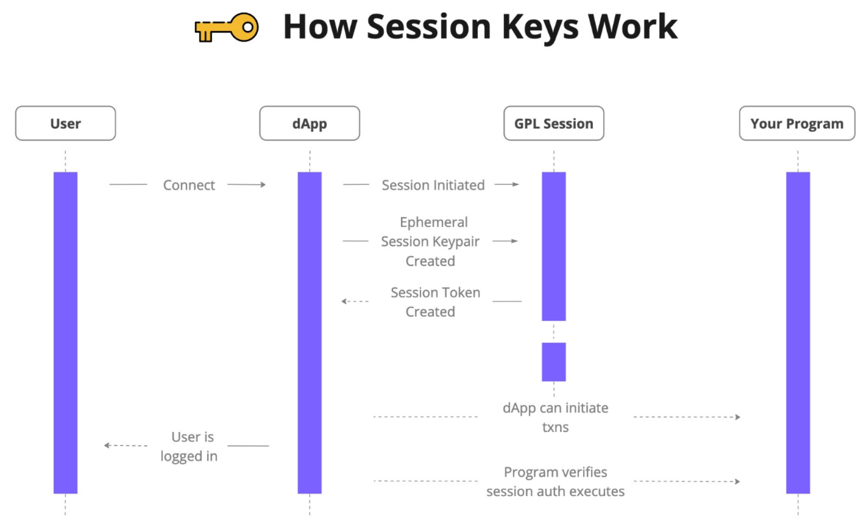
- **Enhanced security**: Batch transactions ensure atomicity, meaning that either all sub-transactions within the batch succeed or none of them do. This provides increased security compared to individual transactions, which could be executed in a non-sequential manner, potentially creating vulnerabilities.
- **Time-saving**: Performing multiple individual transactions may require users to wait until all transactions are confirmed on the blockchain before proceeding with the next transaction. This will cost users more time and money. Meanwhile, Batch Transaction helps pack multiple transactions into a single transaction, saving users time and not having to wait for confirmation before proceeding with the next transaction.

3.3. Delegation

This feature is currently under development, by seeing its potential, we will (absolutely) support this feature in the nearest future.

Session Keys are a massive leap forward for user experience. They allow users to pre-approve an application's transactions according to a set of parameters: a given duration, a max amount of gas, a max transaction volume of a certain token, or a particular function on a particular contract.

Figure 10: How session keys work



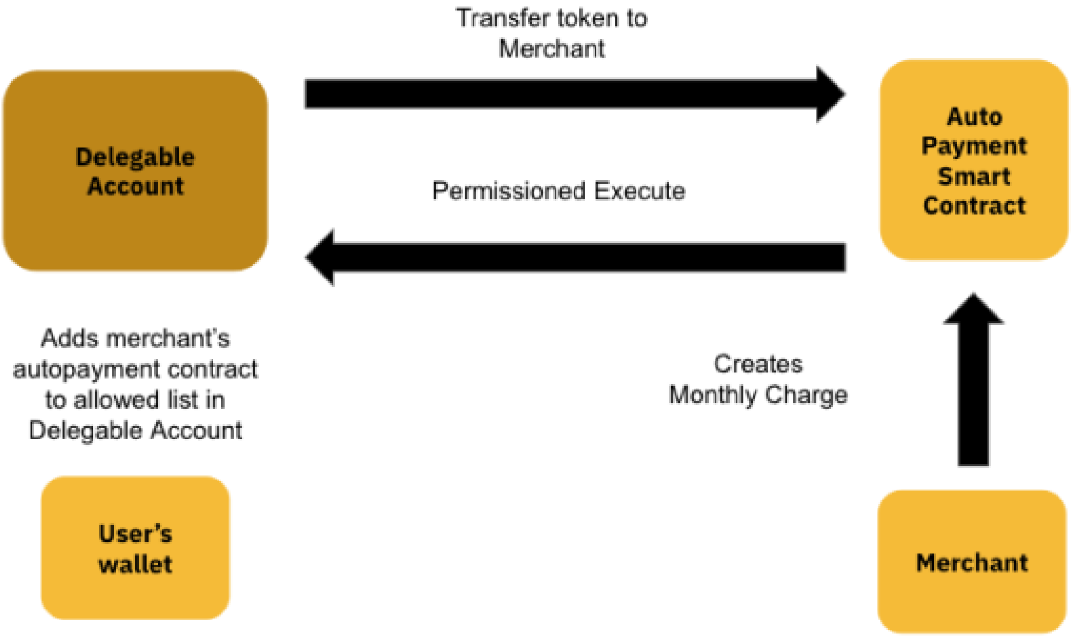
By issuing a session key for a specific dApp, we delegate the dApp the authorization to approve tokens, execute transactions within the dApp's scope, and update the game state without requiring multiple signatures (combined with batch transactions). This will result in a highly favorable user experience.

Within the scope of the whitepaper, I will address various use cases and their operational mechanisms.

Automatic Payments

Instead of having to request payments each time on a blockchain, the wallet can be set up to enable **recurring payments** based on predetermined conditions. While still enjoying full control over the wallet, the user can approve automated, programmable payments for utility and subscription bills.

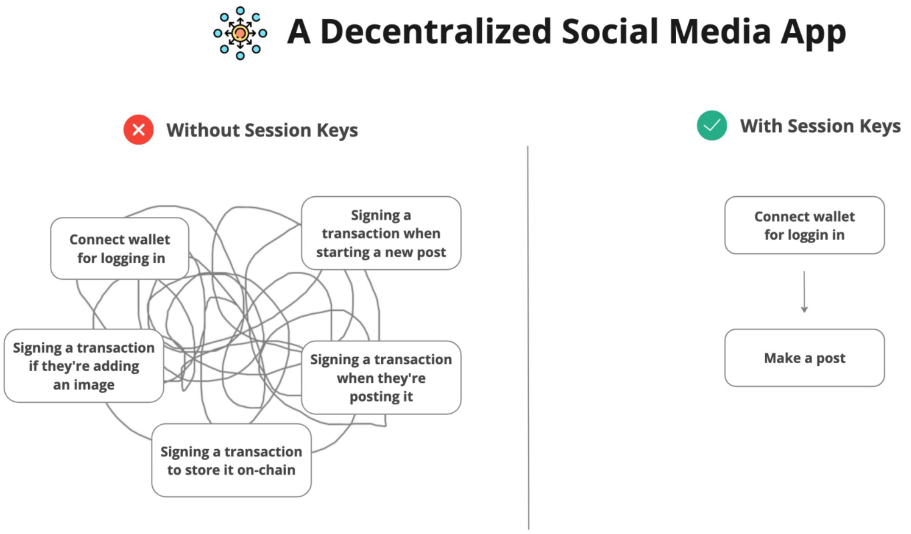
Figure 11: Schematic representation of delegable account design


Transfer token to Merchant
Delegable Account
Auto Payment Smart Contract
Permissioned Execute
Adds merchant's autopayment contract to allowed list in Delegable Account
Creates Monthly Charge
User's wallet
Merchant

Social Media

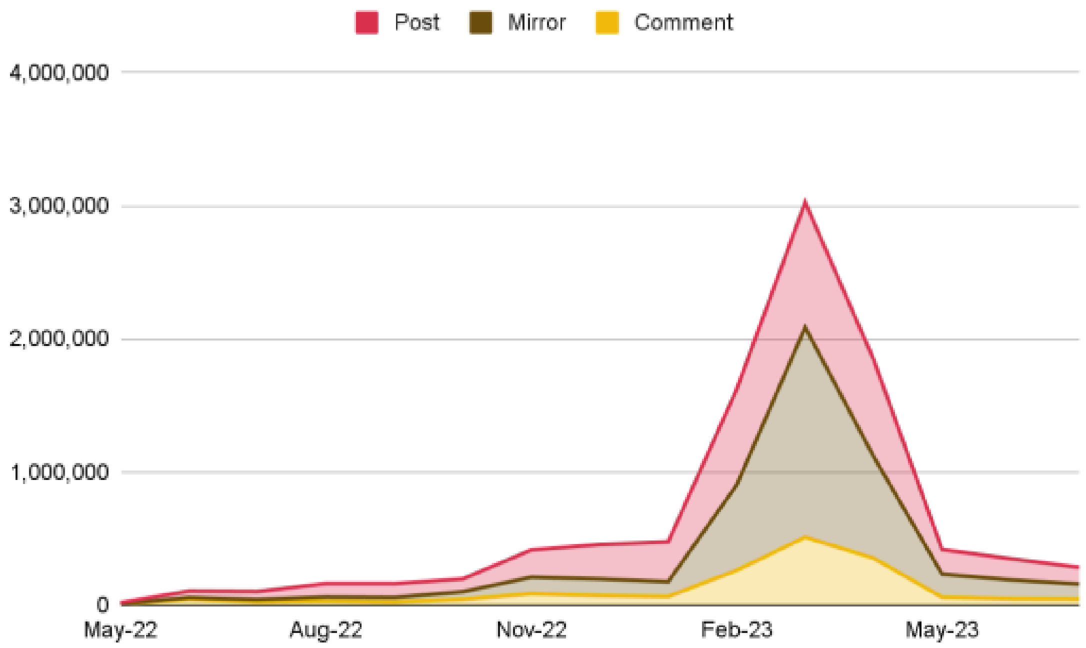
Decentralized social media network **Lens Protocol** has implemented AA via the **dispatcher**.
It provides a tool for users to delegate signing privileges to this dispatcher wallet for functions such as **posting**, **commenting**, and **changing profile metadata**.

Figure 12: A decentralized social media app


A Decentralized Social Media App
✕ Without Session Keys
Connect wallet for logging in
Signing a transaction when starting a new post
Signing a transaction if they're adding an image
Signing a transaction when they're posting it
Signing a transaction to store it on-chain
✓ With Session Keys
Connect wallet for loggin in
Make a post

This enables users to continuously interact with dApps without needing to constantly approve each time. Simultaneously, the **dispatcher** also **pays the gas fees** for these transactions, removing the need for users to hold the native tokens for in-app interactions. Over the past year, the protocol has enjoyed a rise in adoption.

Figure 13: Daily social media activity on Lens Protocol



3.4 Third parties Integration

DApp Integration

We created an SDK version 0.1.0 with the purpose of facilitating the seamless integration of dApps with our wallet. Additionally, we established a dApp marketplace to display all the dApps that have been incorporated into our system. In the upcoming phases, we have intentions to enhance our SDK to enable dApps to utilize more advanced functionalities within our wallet, including subscription services, session keys, and more.

Payment Gateway Integration

This feature is currently under development, by seeing its potential, we will (absolutely) support this feature in the near future.

To facilitate our users in buying crypto with fiat, W3Smart Wallet plans to integrate with payment gateways such as Alchemy Pay (ACH), Banxa,...

B. Notable Developments

Emerging Layer 2 (L2) chains have emerged as feasible alternatives for developers to explore AA functionality. These options encompass:

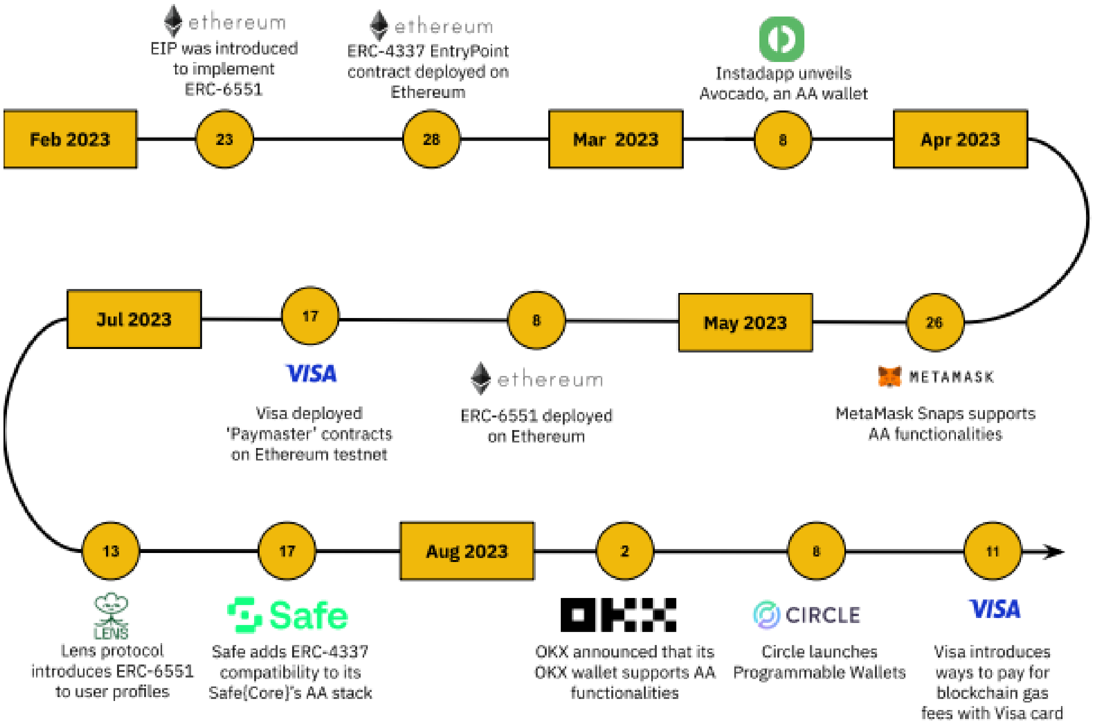
- zkSync emerged as the pioneer EVM-compatible chain to integrate native AA at the protocol level. This entails mandating all accounts to adopt the IAccount interface, which is fully programmable and supports diverse customizations.
- Akinly, the zk-rollup solution, StarkNet, has also integrated AA capabilities. Notably, Argent, utilized by 80% of StarkNet users, serves as an exemplar. Additionally, Visa's experiments involving delegable accounts and automated payments were conducted on this chain.
- Optimistic rollups, exemplified by Optimism and Coinbase's Base, have similarly incorporated AA variants. They furnish developers with APIs to forge novel solutions, incorporating features like Social Onboarding and gasless transactions. Notably, the Base, Safe, and Gelato teams recently introduced bounties at **ETHDenver** to **incentivize** projects integrating AA into their endeavors.

Figure 14: AA-related Ethereum Improvement Proposals

	Features	**Status**
EIP-2771	AA-using meta transactions that allow third parties to pay for a user’s gas costs without making changes to the Ethereum protocol	**Active** Implemented by Biconomy
EIP-2938	Update the Ethereum protocol by introducing a new transaction type to support AA	**Inactive** Inertia for fundamental changes to the EIP-3074 protocol’s consensus
EIP-3074	Upgrade EOA to incorporate smart contract logic into the account	
EIP-4337	AA without requiring changes to the Ethereum protocol. Adds a new system and introduces user operations	**Active**
EIP-6551	Introduces non-fungible TBA, an interface for smart contract accounts owned by NFTs	**Active**
EIP-6900	This modular approach splits account functionality into three categories, implements them in external contracts, and defines an expected execution flow from accounts.	**Active**

We are actively working to enhance the contract according to the EIP-6900 standard, aiming to establish a plugin market for wallets. This will enable effortless addition, removal, and upgrading of functionalities as desired, offering a highly customizable user experience.

Figure 15: Timeline of AA adoption


Feb 2023
23
ethereum
EIP was introduced to implement ERC-6551
28
ethereum
ERC-4337 EntryPoint contract deployed on Ethereum
Mar 2023
8
Instadapp unveils Avocado, an AA wallet
Apr 2023
26
METAMASK
MetaMask Snaps supports AA functionalities
May 2023
8
ethereum
ERC-6551 deployed on Ethereum
17
VISA
Visa deployed 'Paymaster' contracts on Ethereum testnet
Jul 2023
13
LENS
Lens protocol introduces ERC-6551 to user profiles
17
Safe
Safe adds ERC-4337 compatibility to its Safe{Core}'s AA stack
Aug 2023
2
OKX
OKX announced that its OKX wallet supports AA functionalities
8
CIRCLE
Circle launches Programmable Wallets
11
VISA
Visa introduces ways to pay for blockchain gas fees with Visa card

C. Conclusion

The concept of abstraction seeks to simplify the intricacies associated with wallets and blockchains, offering a recognizable interface for Web3 dApps. With a strong emphasis on user experience, this approach significantly facilitates user adoption and presents an appealing avenue for developers to create a seamless and recognizable user interaction.

Undoubtedly, incorporating programmable logic into wallets presents boundless opportunities for developers to enhance the capabilities offered to users by these contracts. The increasing interest in this research domain is reassuring, as indicated by the rise in on-chain statistics, demonstrating the escalating acceptance and inclination of users toward employing these technologies.

D. Acknowledgements

We thank our friends and peers for assistance in conceptualizing, reviewing, and providing support for our work with **W3Smart Wallet.**

- Duong Tran Thi Thuy Hằng Lưu Thu has revisited the content and reformatted the substance of the whitepaper.
- Nhật Cao Huy Nguyen Quang provided feedback on the sections pertaining to technical matters.

E. References

1. https://eips.ethereum.org/EIPS/eip-4337
2. https://eips.ethereum.org/EIPS/eip-6551
3. https://usa.visa.com/solutions/crypto/paying-blockchain-gas-fees-with-card.html
4. https://usa.visa.com/solutions/crypto/auto-payments-for-self-custodial-wallets.html
5. https://usa.visa.com/solutions/crypto/rethink-digital-transactions-with-account-abstraction.html
6. https://ethereum.org/en/roadmap/account-abstraction/
7. https://vitalik.ca/general/2021/01/11/recovery.html