Account Abstraction

Introduction

To interact with the Ethereum blockchain, users need an externally owned account (EOA) with a non-zero ETH balance to pay for gas fees. This preliminary setup introduces two significant disadvantages that affect users' experience with the blockchain:

1. Users must keep their EOAs safe, which is notoriously difficult to accomplish. EOAs are essentially a public-private key pair: the public key represents the identity of the EOA on the blockchain, and the private key is used to manage the funds associated with the account. The private key is necessary and sufficient, meaning that losing the private key makes it impossible to gain back access to the funds, and anyone who gains access to the private key has complete control over the funds associated with it.
2. Users must ensure their ETH balance is enough to pay gas fees for their transactions regardless of what assets they want to interact with. This is equivalent to being required to spend some USD whenever you make a transaction with your credit card in your home country of Japan.
3.

Account Abstraction is a solution to improve the user experience when interacting with the Ethereum blockchain by increasing flexibility and adding more security to the users' accounts. The purpose of account abstraction is toabstract away the technicalities of operating a blockchain account to provide a user experience similar to the one existing in Web2 applications.

Account Abstraction can bring about various features such as transaction limit, key rotation and revocation, automated payments, non-ETH gas payments, batch transactions, and trusted sessions.

Account Abstraction Solutions

The two types of accounts existing on Ethereum today are:

1. EOAs
2. : active accounts that can initiate transactions that change the Ethereum state, but their capabilities are limited to sending ETH or interacting with contracts. EOAs are operated by users through the private key.
3. Contract accounts
4. : passive accounts controlled by the logic in their code. Contract accounts can execute arbitrary logic, but they can only do so in response to a transaction initiated by an EOA.
5.

An ideal candidate for a user-friendly blockchain account should have the flexibility of a contract account with the leverage of an EOA. Naturally, the first general approaches for implementing Account Abstraction were upgrading EOAs to make them programmable or upgrading smart contracts to allow them to initiate transactions. However, both these approaches require protocol changes that would result in a hard fork and are unlikely to be adopted soon.

EIP-4337 introduced a more feasible approach to Account Abstraction that relies on higher-layer infrastructure instead of protocol changes. EIP-4337 introduces theUserOperation object representing a transaction to be sent on behalf of a user.UserOperation objects are sent to a dedicated operation mempool and are handled by specialized actors called bundlers.

Given that Account Abstraction is an essential step for the mass adoption of Ethereum, other approaches were proposed, and below, we introduce our own proposal, "Seedless Authentication".

Seedless Authentication

Seedless authentication aims to simplify the user experience in the blockchain space by eliminating the need to manage cryptographic seeds. It uses Safe multi-sig wallets and FIDO2 devices to allow users to create a blockchain account and send transactions with their biometrics only. With seedless authentication, each user account is a multi-sig smart contract that can be programmed to employ multi-factor authentication and key recovery and revocation.

The smart contract is controlled by one or multiple keys (depending on how the user sets up the account) that reside in FIDO2 devices. A FIDO2 device is a device that allows a user to easily authenticate by using their biometrics; the device generates a public-private key pair - the private key is stored on the device and never exposed to the user, and the public key is stored in the smart contract that represents the user's blockchain account. When the user wants to send a transaction, the device will create a signed object that the smart contract will validate and execute the transaction only if all the checks pass. The beauty of it is that the user does not need to handle any private key (the FIDO2 device manages it); all the user needs to remember is the smart contract address. Moreover, the private key is securely stored inside the device and can only be accessed when presenting the user's biometrics.

All users' seedless authentication interactions account go through a Relayer. The Relayer will deploy the Safe multi-sig smart contract when the user creates the account and will send the transaction containing the authentication object created

by the FIDO2 device to the contract; the Relayer pays all the gas fees associated with these operations as well.

Conclusion

Account Abstraction is a proposal for improving the flexibility and security of managing user accounts on the Ethereum blockchain by eliminating, among other things, the need to store private keys or seed phrases and pay gas fees. The initial approaches require protocol changes and have yet to be adopted, but solutions that do not rely on protocol changes are also emerging. Seedless authentication is one such solution, a design that allows users to use the blockchain with the touch of their fingertips... literally!

References

1. Account Abstractionhttps://ethereum.org/en/roadmap/account-abstraction/
2. FIDO2https://fidoalliance.org/fido2/
3. Account Abstraction EIPs:https://eips.ethereum.org/EIPS/eip-2938
4. ,https://eips.ethereum.org/EIPS/eip-3074
5. ,https://eips.ethereum.org/EIPS/eip-4337
6. ,https://eips.ethereum.org/EIPS/eip-5003
7.

Previous Carrier Next Decentralized Randomness Last updated6 months ago On this page *Introduction * Account Abstraction Solutions * Seedless Authentication * Conclusion * References

Was this helpful?