Account Abstraction

ERC-4337 Compatible Bundler & Paymaster Users interact with Ethereum using externally owned accounts (EOAs) which are public-private key pairs. Anyone with access to the private key can perform actions within the rules of the Ethereum Virtual Machine (EVM). By design, the Ethereum network can only go through state transition when an EOA triggers a transaction and consequently pays a gas fee in ETH. These factors limit how users can interact with the blockchain.

1. Poor security
2.
    1. Lost private keys cannot be recovered
3.
    1. Compromised private keys give attackers instant access to all funds in the account
4.
    1. Rigid security rules (e.g., must use ECDSA
5.
    1.)
6. 5.
7. Lack of customization
8.
    1. Must initiate or sign every transaction
9.
    1. Not programmable (i.e., can not define custom rules
10.
    1.)
11. 4.
12. Gas payment
13.
    1. Account must hold ETH at all times in order to cover transaction fees
14.
    1. Can not use other tokens (e.g.,
15.
    1. ERC20
16.
    1.)
17. 5.
18.

Smart contract wallets are the solution to these problems by allowing users to flexibly program better security and user experiences into their accounts. Account abstraction enables smart contracts to initiate transactions themselves, without the user having to manage a separate EOA and ETH balance. This opens up the door to many exciting use cases.

What is a Smart Wallet?

Smart wallets are wallets controlled by smart contracts following theERC-4337 specification . Ethereum has two types of accounts:

1. Externally Owned Accounts
2. (EOAs)
3. Contract Accounts
4. (Smart Contracts)
5.

A Contract Account is managed by a Smart Contract rather than an EOA and relies on code instead of private keys to secure and recover wallet information.

Benefits of Smart Wallets

1. Fully programmable
2.
    - Can do anything a smart contract can
3.
    - Upgradeable to add new features
4. *
5. Arbitrary verification logic & recovery
6.
    - ECDSA (EOA controller account)
7.
    - Social Login

8.
    - Session Keys
9.
    - Biometric
10. *
11. Atomic multi-operations
12.
    - Combine multiple transactions into a single atomic transaction
13.
    - Better efficiency as call overhead is only incurred once
14.
    - E.g., approve & spend tokens
15. *
16. Gasless transactions
17.
    - Fully sponsored &ERC-20
18.
    - payment
19.
    - Compatible with all smart contracts out of the box
20. *
21. Semi-abstracted nonces
22.
    - Concurrent execution channels
23. *
24.

Why ERC-4337?

Unlikeother proposals , ERC-4337 avoids changes to the consensus layer itself increasing the chance of faster adoption.

Terminology

Sender

The sender is an ERC-4337 compatible smart contract wallet storing the users assets.

It must implement the following interface:

```

Copy interfaceIAccount{ functionvalidateUserOp( UserOperationcalldatauserOp, bytes32userOpHash, uint256missingAccountFunds )externalreturns(uint256validationData); }

```

UserOperation

AUserOperation is a pseudo-transaction object sent by the user into an alternate mempool.

It contains the following fields:

Field Type Description sender address Account requesting the operation nonce uint256 Anti-replay parameter initCode bytes Account creation code (only required if not yet created i.e., first transaction) callData bytes Data passed tosender during execution callGasLimit uint256 Gas allocated for main execution verificationGasLimit uint256 Gas allocated for verification preVerificationGas uint256 Amount allocated to compensate the bundler for any gas overhead not tracked on-chain by theEntryPoint maxFeePerGas uint256 Similar toEIP-1559 maxPriorityFeePerGas uint256 Similar toEIP-1559 paymasterAndData bytes Paymaster address andcallData (empty for self-sponsored transactions) signature bytes Data passed to the account along with the nonce during the verification step

EntryPoint

TheEntryPoint is a singleton smart contract that handles the verification and execution of bundles ofUserOperations . This ensures much of the complicated logic is not required in the wallet itself and Instead, wallets trust theEntryPoint to perform proper validation (similar to atrusted forwarder).

Bundler

A bundler is a node that bundles together multipleUserOperations from an alternate mempool and forwards them to theEntryPoint contract as a single transaction. The bundler executes transactions via EOAs which cover the transaction fees

upfront and are later compensated. TheGelato Bundler is built on top of the existingGelato Relay service and sponsors transactions via1Balance .

SeeAdvantages & Highlights .

Paymaster

A paymaster is a service that covers transaction fees on behalf of the user. Unlike other solutions, Gelato does not rely on the on-chainEntryPoint to compensate transaction costs. Instead, fees are settled by the1Balance paymaster post-execution which avoids overcharging users and reduces the overall on-chain footprint.

SeeAdvantages & Highlights .

Previous Subscription & Payments Next Supported Networks Last updated4 months ago On this page *What is a Smart Wallet? * Benefits of Smart Wallets * Why ERC-4337? * Terminology * Sender * UserOperation * EntryPoint * Bundler * Paymaster