**8 Benefits of Smart Contract Wallets**

Discover the UX and Security Benefits of Wallets that Use Account Abstraction (ERC-4337)

**Last Updated:**

June 1, 2023

Smart contract wallets that use [**ERC-4337**](https://www.alchemy.com/overviews/what-is-account-abstraction) are web3 wallets that manage smart contract accounts, and they offer the simple user experience of conventional EOA (Externally-Owned Account) wallets, with additional functionality, flexibility, and security benefits.

## What are the benefits of smart contract wallets?

Smart contract wallets leverage the programmability of smart contracts to increase their utility compared to EOA wallets. Although [**smart contract wallet transactions need to be initiated by EOA wallets**](https://www.alchemy.com/overviews/how-do-smart-contract-wallets-work), Account Abstraction infrastructure services like Bundlers, Paymasters, the EntryPoint contract, and Wallet SDKs make the process simpler for wallet developers and end users.

**Some of the possible features enabled by smart contract wallets include:**

1. Two-factor authentication
2. Social Recovery
3. Flexible gas policies
4. Custom Signature Schemes
5. Multicall
6. Spending Limits
7. Allowlisting addresses
8. Multi-sig wallet support

### 1. Two-factor Authentication

[**Smart contract wallets**](https://www.alchemy.com/best/smart-contract-wallets)can offer two-factor authentication, which requires two different components to confirm a user’s identity, for example:

* something the user knows, such as a password;
* something the user has, such as a phone;
* something the user is, such as a fingerprint.

This is a well-known web2 feature that provides an added layer of security and prevents unauthorized access.

### 2. Social Recovery

Smart contract wallets are programmable, and during social recovery the public key of a smart contract wallet is changed if the original private key was lost by the wallet owner. This is achieved through a multisig transaction with predefined trusted friends or relatives (known as guardians). This removes the need for the owner to store seed phrases.

### 3. Flexible Gas Policies

Smart contract wallets can create flexible gas policies using APIs that support ERC-4337's Paymaster specification. Being able to [**manage gas policies**](https://www.alchemy.com/account-abstraction) enables developers to sponsor transactions on behalf of users (i.e. gasless transactions) and allow users to pay for gas in any ERC-20 tokens (e.g. USDC).

#### Sponsoring Transactions

The transaction would contain a designated sponsor of the fee in its **calldata**. The Paymaster would check with the sponsor if it is willing to [**sponsor the transaction fee**](https://docs.alchemy.com/docs/gas-manager-services) and, if so, proceed with the execution of the transaction.

#### Paying for Gas with Any ERC-20 Token

The Paymaster also allows for fee payment in currencies other than the blockchain's native token.

For example, if a wallet owns only an ERC-20 token like USDC and not the native token on Ethereum (ETH), the Paymaster would check if the USDC balance is enough to cover the fee. If the wallet has enough USDC, the Paymaster will pay for the transaction execution with ETH and accept the wallet's USDC as compensation.

### ****4. Custom Signature Schemes****

Smart contract wallets could easily be programmed to use a [**different signature scheme**](https://www.alchemy.com/blog/account-abstraction-aggregate-signatures) to the default ECDSA used in Ethereum. This default signature scheme could be substituted with multisig, social recovery, or even a quantum resistant option like Crystals-Kyber.

### ****5. Multicall****

Multicall is the ability of a smart contract wallet to execute more than one action in a single atomic transaction. Multicall functionality improves the current wallet paradigm where every transaction is propagated and confirmed separately, resulting in longer processing time and poor user experience.

For example, if a user wants to trade a token on Uniswap, they must first click ‘approve’ and wait for the transaction to be confirmed. Afterwards, they must wait for the ‘swap’ transaction to be confirmed, and only then is the token exchange completed.

Smart contract wallets would allow these transactions to be batched into one atomic transaction.

### ****6. Spending Limits for Different Signers****

Smart contract wallets can be set to recognize more than one valid signer, which could be used to set different spending limits for different signers. For instance, a user could add both their laptop and Ledger hardware wallet as signers, allowing transactions of up to 0.05 ETH through their laptop, but requiring that the hardware wallet be used for larger transactions.

### 7. Allowlisted and Denylisted Addresses

Smart contract wallets allow users to grant or deny one or more wallet addresses access to their account. Allowlisting enables the user to grant access, while denylisting accounts would block or deny accounts access. Both of these features improve security and give users additional control over their account.

### 8. Multi-sig Wallet Support

The multi-sig support feature of smart contract wallets is particularly important for organizations, which often require more than one approval before a transaction can be executed. Multi-sig wallet support increases the security of transactions by requiring multiple people to authorize them, which makes it hard for a single person to steal funds or execute unauthorized transactions.

## What are the disadvantages of smart contract wallets?

Some of the main drawbacks of smart contract wallets are: higher gas costs, potential code issues, and the current lack of support by most web3 platforms.

### 1. Higher Gas Costs

Smart contract wallets are controlled by smart contract code, and it’s necessary to pay for the computational resources required to execute this code. Executing smart contract wallet transactions are more expensive than operating an EOA wallet, especially if there are additional features that require complex smart contract code to run.

### 2. Potential Code Issues

Smart contract wallets are code-based, which means they face many of the same auditing, security, and bug issues as an average software product. Because smart contract wallets are new, users adopting SC wallets should evaluate the team, wallet design, and code security of new ERC-4337-compliant wallets.

### 3. Lack of Support

Smart contract wallets are still in their early phase and are only supported by a few platforms. For smart contract wallets to go mainstream, they need to be supported by top-tier DeFi protocols and other web3 platforms.

## Should I use smart contract wallets?

The main reasons why one should consider using smart contract wallets are: improved security features, automation, decentralization and user-friendliness.

1. **Security** - 2FA, multi-sig support, and social recovery make it safer to use than EOAs
2. **Automation** - programmatically interact with web3 protocols and smart contracts
3. **Friendly** **UX** - SCWs eliminate the complexity associated of EOA wallets