I would like to open the discussion about a proposal for a new opcode named IMPERSONATECALL that calls other contracts and replaces the msg.sender at the same time. It saves gas and simplifies several use cases regarding meta-transactions and sponsored wallets.

You can read the proposal here:

github.com

ethereum/EIPs/blob/716fadffb103dd9e491d7ca6cb98b3f98036ea71/EIPS/eip-IMPERSONATECALL.md

---

eip: title: IMPERSONATECALL Opcode author: Sergio Demian Lerner (sergio.d.lerner@gmail.com) category: Core type: Standards Track status: Draft created: 2020-09-24

Overview

Add a new opcode, `IMPERSONATECALL` at `0xf6`, which is similar in idea to`CALL`, except that it impersonates a sender, i.e. the callee sees a sender different from the real caller. To prevent collisions with other deployed contract or externally owned accounts, the impersonated sender address is derived from the real caller address and a salt.

Specification

`IMPERSONATECALL`: `0xf6`, takes 7 operands:

- `gas`: the amount of gas the code may use in order to execute;
- `to`: the destination address whose code is to be executed;

This file has been truncated. show original

The idea is that a contract can impersonate child contracts (created with a derivation similar, but not equal, to CREATE2). Therefore there is no practical risk that the caller impersonates a third party contract.

This opcode enables the creation of multi-user wallets, where each user is given a separate non-custodial smart-wallet having its own address for storing ethers and tokens, yet no contract code is deployed, and a main-wallet contract retains the common functionality (i.e. social private key recovery). Wallets are accessed by a meta-transaction system (i.e using EIP-712) embedded in the multi-user wallet contract.

Even if the same functionality can be achieved by using counterfactual contract creation, this solution is attractive because:

- It’s much simpler to design and less error prone.
- It provides the sponsor huge gas savings, removing the need for the deployment of thousands of wallets.

I’m sure there are plenty more use cases that can benefit from this opcode.