Design Options for Web3 Non-Custodial Cryptocurrency Wallet based on Web2 Authentication

Author: Bc. Adam Maczkó
Supervisor: doc. Ing. Ivan Homoliak, Ph.D.

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2025

Abstract

The rising popularity of blockchain technology has led to unprecedented growth in the use of cryptocurrencies and decentralised applications. Yet newcomers still struggle with the complexity of cryptocurrency wallets. Many first-time users therefore rely on custodial wallets provided by centralised exchanges, thereby surrendering control over their private keys. This work presents design options for a wallet that delivers the usability and familiarity of those custodial services while remaining fully non-custodial, ensuring that users keep exclusive ownership of their keys without facing a steep learning curve.

Design and key storage options

- **Embedded wallets** – cryptocurrency wallets that live directly inside the dApp.
- **Browser-extension wallets** – add-on containers that inject a provider into web pages (*e.g.*, MetaMask-style).
- **Hardware wallets** – external USB or NFC devices that keep keys fully offline.
- **Smart Contract Wallets** (ERC-4337 Account Abstractions) – can be programmed to enforce custom policies.
- **Web 2.0 authentication** – password, OIDC, or passkeys to simplify onboarding.
- **Multi-factor authentication (MFA)** – TOTP, hardware security keys, or biometrics layered on top of wallet access.
- **Different private key storage schemes** (Fig. 1–2).

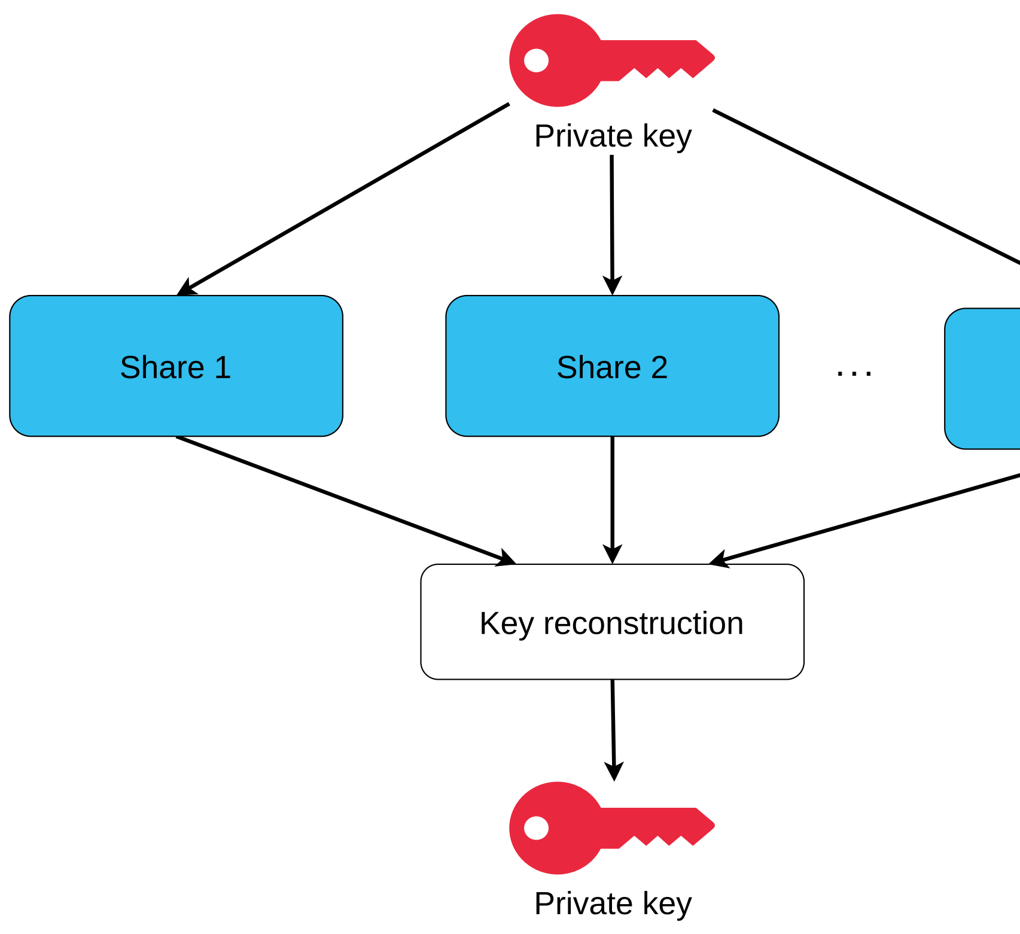

Figure 1: Shamir's Secret Sharing.

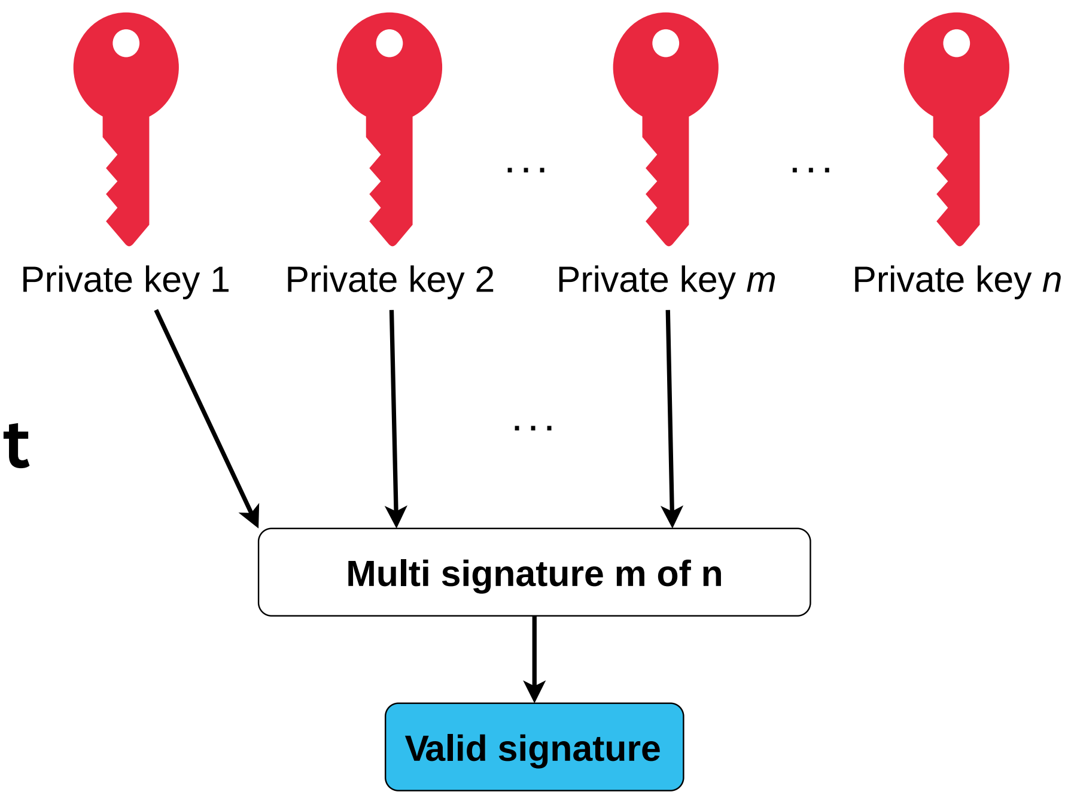

Figure 2: Multi Signature m of n scheme.

Proposed wallet architecture

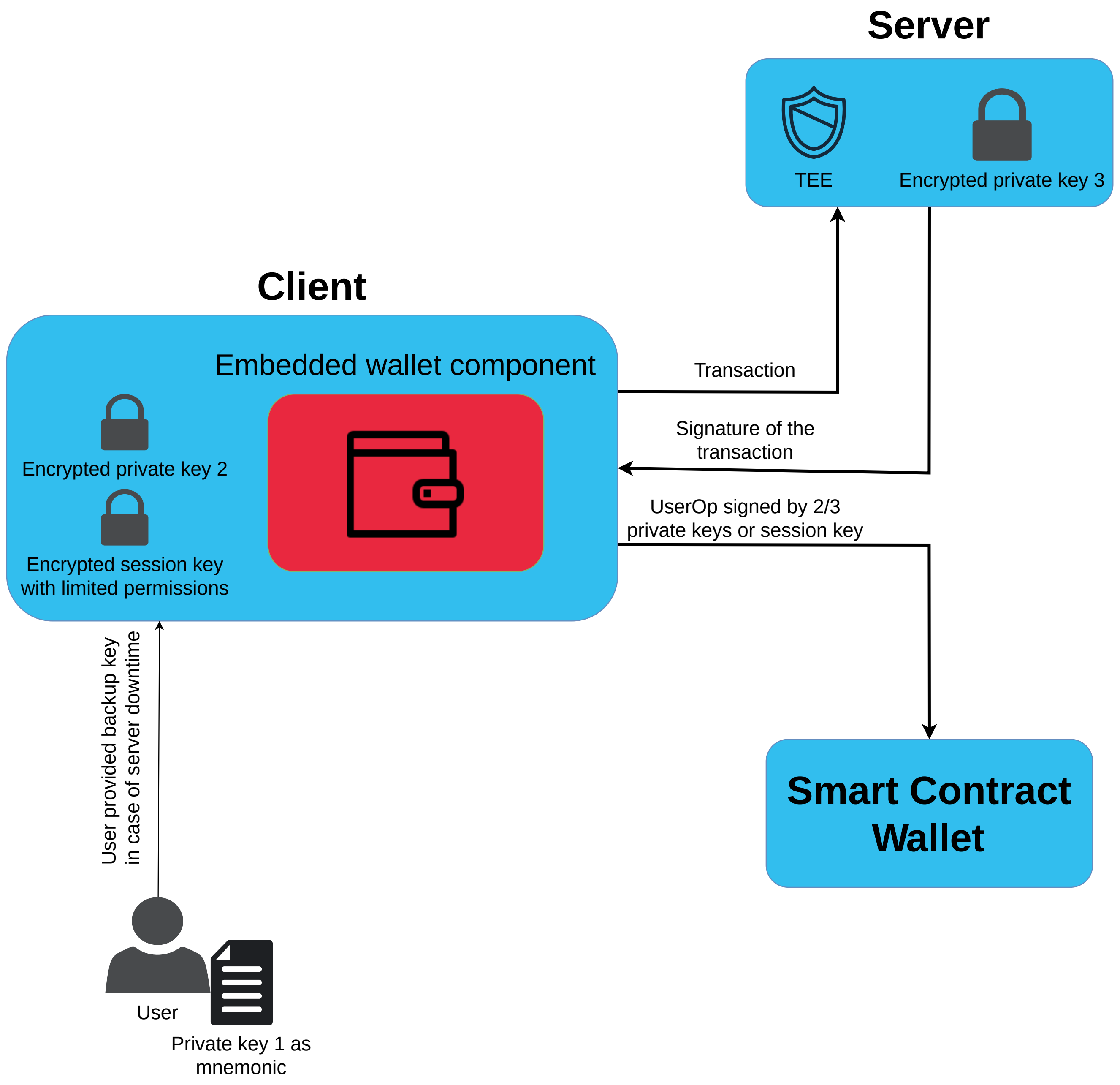
Server
TEE
Encrypted private key 3
Client
Embedded wallet component
Encrypted private key 2
Encrypted session key with limited permissions
Transaction
Signature of the transaction
UserOp signed by 2/3 private keys or session key
Smart Contract Wallet
User provided backup key in case of server downtime
User
Private key 1 as mnemonic

Figure 3: Non-custodial embedded wallet based on 2 of 3 multi signature.

- **2-of-3 threshold:** mnemonic (user), encrypted browser key, encrypted TEE key
- OIDC login with optional passkey
- Client sends unsigned transaction over TLS
- TEE returns partial signature; client adds its own
- UserOp executes in smart-contract wallet only with two signatures
- Session keys with spend limits permit single-signature micro-payments
- Larger transfers demand full 2-of-3 approval
- Server controls one key share, so it can't block or steal funds