Security Challenges in HSM-Based Wallets — and How Split-ECDSA (SECDSA) Solves Them

ITU Workshop 30 March 2026

Session 4: Security of wallets

Eric Verheul — Eric.Verheul@wellet.nl

Wellet

The wallet done well

wellet.nl

Dutch/German HSM-Based Wallet Challenges & How SECDSA Solves Them

Challenge		SECDSA Solution
PIN-data leakage User authentication data is processed outside the HSM allowing PIN brute-force attacks.	→	**HSM-bound PIN verification** User PIN checking is done inside the HSM using standard PKCS#11 functionality.
Repudiation risk Consequently, User authentication data cannot be securely stored in the wallet Transaction Log — user consent cannot be transparently proven.	→	**Publicly verifiable user authentication** User requests towards wallet provider are publicly verifiable by any party, eliminating repudiation risk. Also solves remote signing issue going back to 1999.
Need for custom HSM firmware (vendor lock-in) Mitigating PIN–data leakage forces bespoke HSM firmware — limiting scalability, flexibility and introducing HSM vendor lock-in. HSM-firmware still not addresses the repudiation risk.	→	**No custom HSM firmware needed (no vendor lock-in)** Wallet Provider component handling HSM ('WSCA') can run as standard software (e.g. containerized) — no custom HSM firmware required. Also works with eIDAS certified HSMs ('EN 419221-5') for optimal compliance.
Multiple HSM calls → poor scalability Each user instruction requires multiple HSM interactions, e.g., secure messaging, verification key decryption, PIN-validation.	→	**1 HSM call overhead per user instruction** Only one standard HSM ('PKCS#11') call per user instruction request.