Perry's Résumé - Custody & Monetary Systems

Perry Kundert <*perry@kundert.ca*> +1-780-970-8148

2026-08-21 12:00:00

R&D engineer with 30+ years building systems where a wrong answer is expensive or dangerous; the last five spent on key management and monetary infrastructure. Shipped a Shamir secret-sharing custody application (SLIP-39: group-threshold seed backup with air-gapped ceremony, signed macOS/win32 apps) and designed a complete wealth-backed monetary platform (EVM-implemented): identity-aware ledgering with on-chain credit-limit enforcement against insured assets, a privacy-preserving bearer-note layer on Groth16 SNARKs, and PID-controlled issuance – proven with agent-based economies run for simulated years on a deterministic EVM simulation harness, every architectural ruling captured in written decision records. AI-native by daily practice: orchestrated agent fleets, multi-model review, and a production LLM service with an order-of-magnitude serving cost reduction. (PDF, Text)

Technical Skills

- **Key Management & Custody**: SLIP-39 / Shamir secret sharing (group-of-groups thresholds), BIP-39 seed backup and recovery, HD wallet derivation (Trezor/Ledger-compatible), air-gapped generation ceremonies, BIP-38/JSON encrypted paper wallets, printable recovery-card PDFs
- **Smart Contracts & EVM**: Solidity (ERC-20, ERC-721, custom), Ethereum, Polygon, Hardhat, Foundry, OpenZeppelin, DeFi integration (Uniswap AMM pools, Chainlink oracles)
- **Zero-Knowledge & Cryptography**: Groth16 SNARKs, circom/snarkjs, Poseidon hashes, Merkle trees, nullifier sets, ElGamal encryption, Chaum-Pedersen equality proofs, Schnorr signatures, Pointcheval-Sanders credentials, Fiat-Shamir NIZK transforms; implemented, composed, and formally verified on BN254
- **AI-Native Engineering**: coding-agent fleet orchestration, multi-model review workflows, LLM serving (Claude API / Ollama / MLX) behind one contract, KV-cache optimization, local inference operations (llama-cpp / Ollama / MLX)
- **Languages**: Rust (5+ yrs), C++ (25+ yrs), Python (15+ yrs), C, Go, Solidity
- **Web & Browser**: JavaScript/WASM (browser crypto kernels, EVM simulation clients, typed API definitions); TypeScript-adjacent – Next.js not yet; learns stacks quickly, and in public
- **Distributed Systems & Consensus**: CAP/BFT systems, partition-tolerant protocol design, DHTs, agent-centric accounting, consensus without global coordination
- **Networking & Concurrency**: JSON-RPC, gossip protocols, mTLS 1.3, WebSockets, WebTransport, QUIC, lock-free wait-free concurrency, real-time systems
- **DevOps & Infra**: Linux, Docker, CI/CD, Git, deterministic builds (Nix), GDB, Valgrind, Linux Perf

Experience

R&D Consultant, *Dominion R&D Corp.* (Remote, 2009–Present)

Architected a complete blockchain-based monetary system; smart contracts, cryptographic identity, zero-knowledge privacy primitives, and economic mechanism design. Guided teams through cryptographic and distributed systems architecture.

- Designed the **Alberta Buck** protocol, a full Ethereum EVM smart contract system comprising:
  - **BUCK** (ERC-20): Identity-aware fungible token with continuous demurrage accounting, on-chain credit-limit enforcement against pledged assets, and Chaum-Pedersen NIZK re-encryption proofs gating every counterparty-pair transfer.
  - **BUCK_CREDIT** (ERC-721): Insured real-world-asset NFTs with deterministic depreciation curves and parametric insurance integration; the collateral backing the ERC-20 supply.
  - **IdentityRegistry**: On-chain PS-signature credential registry with ElGamal-encrypted identity points, verified via BN254 precompile NIZK at registration; the trust anchor every transfer is gated on.
  - **Notes**: Privacy-preserving bearer-instrument layer. Tornado-style incremental Poseidon Merkle commitment pool with Groth16 SNARK batch-mint and per-flavor spend circuits. Three flavors (A1 addressed cheque, A2 private cheque, B1 bearer cheque) collapse to two circuit shapes covering every deferred-approve use case: self-wormholes, signed cheques, printable paper notes indistinguishable from cash.
  - **BUCK_K**: PID-controlled algorithmic issuance stabilizer referencing a commodity-basket price oracle; automated monetary policy without central bank intervention.
- Proved the platform's stability the way a ledger should be proven (2026):
  - Deterministic multi-backend EVM simulation harness – in-process `pyrevm` at ~7000x per-tx, `anvil`, `tevm` – with byte-identical results across backends.
  - Agent-based economies (honest mortgage refinancers, savers, arbitrageurs, market makers, adversarial whales) run for five simulated years across a ten-configuration experiment matrix: multi-hour autonomous runs, checkpointed telemetry, live progress rendering.
  - Rust/Python/JS/WASM kernels sharing byte-identical test vectors.
  - Two upstream snarkjs PRs: #631 (documentation + test pinning the undocumented Groth16 verifier parameter ABI) and #632 (Nix/Foundry developer-onboarding harness).
  - Written decision records for every architectural ruling, runbooks, and licensing/REUSE diligence.
- Composed and verified the full cryptographic stack with executable `py_ecc` sanity checks on BN254 (formal verification by qualified mathematicians and cryptographers remains to be done):
  - Pointcheval-Sanders signature rerandomization for unlinkable credentials.
  - Chaum-Pedersen re-encryption equality proofs for bilateral identity disclosure; proved completeness, special soundness, and honest-verifier zero-knowledge, lifting to NIZK via Fiat-Shamir in the random oracle model.
  - A-spend and B-spend SNARK soundness, confirming that only the named recipient can deposit an addressed note and that bearer-cheque deposit always identifies the depositor.
  - Domain-separated Poseidon PRF nullifiers with double-spend prevention across both note flavors.
  - Mutual-decryptability invariant preservation; every note spend yields bilateral identity disclosure between counterparties.

- Designed economic mechanisms for MEV-resistant DeFi integration:
  - Analyzed how PID-controlled issuance (with proportional and derivative terms) automatically counteracts whale manipulation of BUCK/stablecoin AMM pools; the system transfers the attacker's assets to legitimate credit holders.
  - Modeled the effect of front-running-resistant oracle design: multiple independent controller implementations (PID, Kalman-filtered PID, MPC) with median-value aggregation prevent any single oracle compromise from moving the issuance multiplier.
  - Designed parametric default insurance with risk-proportional premium investment in a mutual insurance pool; algorithmic premium setting eliminates underwriter discretion.
- Developed widely-used open-source libraries:
  - `ezpwd-reed-solomon`: High-throughput C++/JS Reed-Solomon FEC, used by aerospace and defense systems globally.
  - `cpppo`: Industrial EtherNet/IP protocol implementation in Python.
- Alberta Buck Architecture · BUCK Notes · Formal Proofs · Identity System
- Ethereum Implementation · Slide Deck · ezpwd-reed-solomon

Creator, *SLIP-39 Seed Backup & Recovery* (Remote, 2021–Present)

Shipped multi-party custody to households: no single point of failure for a hardware wallet seed – no burglar, fire, or estate lawyer holds the whole secret.

- Built and ships python-slip39 and the signed SLIP-39 macOS/win32 App (.dmg/.pkg/.zip installers): SLIP-39 Shamir group-of-groups seed generation and backup, with printable recovery-card PDFs sized for real-world storage.
- Air-gapped ceremony by design: seeds are generated and printed on a secure, offline computer, with nothing trusted to disk; recovery is entered directly on-device (Trezor), keeping the seed off every networked machine.
- Backs up existing BIP-39 hardware-wallet phrases (Ledger, etc.) as SLIP-39 card sets, and recovers them round-trip; generates Trezor/Ledger-compatible HD wallet accounts (BTC, ETH, and others) with addresses and QR codes for verification.
- A *significant* enhancement to Trezor's python-shamir-mnemonics; PR #51 provides enhanced recovery of SLIP39 Mnemonic sets in the face of loss, corruption or attack, and ability to alter and re-issue groups.
  - Adds the capability to *replace* SLIP39 mnemonic groups in already-deployed SLIP39 sets. This holds **significant** relevance when employing SLIP39 in business and other organizations.
  - Allows you to *increase* the number of mnemonics in an *existing* multi-mnemonic group; All existing mnemonics remain valid.
- Outputs BIP-38 and JSON encrypted paper wallets for standard software-wallet import.

AI-Native Engineering Practice, *Dominion R&D Corp.* (Remote, 2026)

Agent-orchestrated development as the daily working method, and LLM serving in production.

- **BUCKy**: production LLM question-answering service over a ~100k-token knowledge base; three interchangeable inference backends (Claude API / Ollama / MLX) behind one contract, with the knowledge-base KV cache precomputed and cloned copy-on-write per session – roughly a 10x per-query serving cost reduction; live on perry.kundert.ca.
- Agent-fleet development workflow: orchestrated background coding agents running a ten-configuration agent-based economic simulation matrix – multi-hour autonomous runs, checkpointed telemetry, live dashboard rendering, multi-model code review – with findings written up as decision records while rulings were made interactively.
- Local inference operations: llama-cpp, Ollama, and MLX deployments, prompt/KV-cache management, and a Cloudflare Worker front-end.

Distributed Systems R&D, *Holo Ltd.* (Kelowna, BC 2018–2020)

Architected and tested prototypes of HoloFuel's novel partition-tolerant transaction engine in Rust.

- Implemented agent-centric accounting enabling linear O(n) scaling vs. $O(n^2)$ blockchain bottlenecks; each agent held its own signed transaction chain, with bilateral settlement during network partitions.
- Validated simultaneous atomic transactions across partitioned networks; demonstrated that mutual-credit systems can settle without global consensus.
- Developed statistical models and PID control systems for economic stabilization across distributed agents.
- Contributed to R&D for Holochain's distributed application framework.

Software Engineer, *clearGRID Ltd.* (Kelowna, BC 2017–2022)

Built fault-tolerant distributed telemetry consensus without synchronized state machines.

- Architected real-time SDR system processing 25 Msps I/Q signals on commodity hardware; consensus on meter readings across lossy RF channels without global coordination.
- Optimized AVX/NEON SIMD vector processing for multi-channel demodulation at the network edge.
- Implemented fault-tolerant distributed consensus for partial data collected from multiple samples across unreliable channels.

Sr. IT Advisor, *Enbridge Pipelines* (Edmonton, AB 2002–2009)

Deployed partition-resilient SCADA maintaining safety-critical consensus during network splits.

- Architected Reed-Solomon erasure-coded multi-route protocol enabling real-time SCADA operation through crippling communications failures; a BFT-style availability guarantee implemented a decade before blockchain made the terminology common.
- Implemented multi-path routing maintaining cryptographic integrity across hostile network environments.
- Developed solid-state RTU firmware for 24/7 pipeline control with *<1 defect per 10,000 LOC.*
- Secured communications for $100B+ critical energy infrastructure.

Software Developer, *Hewlett-Packard* (Calgary, AB 1989–1996)

Pioneered state-machine consensus for distributed industrial control systems.

- Re-engineered RTAP's core SCADA alarm system with DFA state machines for geographically distributed safety-critical workflows; deterministic state transitions without central coordination, deployed continent-wide and still operational after 30+ years.

Open Source & Publications

- python-slip39: Cross-platform SLIP-39 Shamir secret sharing implementation (Python); the SLIP-39 macOS/win32 App
- ezpwd-reed-solomon: High-performance Reed-Solomon FEC (C++, JavaScript)
- cpppo: Industrial EtherNet/IP and Modbus protocol implementation (Python)
- holofuel-model: Distributed mutual-credit economic simulation
- Authored series of technical papers on custody, monetary system design, blockchain architecture, zero-knowledge identity systems, cryptographic protocol composition and formal verification, and consensus algorithms; perry.kundert.ca/range/finance/
- Access to pre-release repos on request: alberta-buck

Education

B.Sc. Computer Science, *University of Calgary* (Calgary, AB 1984–1989)