QDAY AEVUM TESTNET (CHAIN 44005 · SHANGHAI EVM)

LAMPORT-SHA256 PQC + EIP-3009 GASLESS

Aevum PQC Invoice Escrow

Post-Quantum verifiable stablecoin (`tUSD8`) invoice escrow & gasless EIP-3009 settlement prototype targeting **QDay Aevum Testnet (Chain ID 44005)**.

15 / 15

Foundry tests PASS (14 unit + 8,192-call invariant, Shanghai EVM)

31 / 31

Python PQC Verifier `pytest -q` PASS (ABI parity + replay guards)

0 H / 0 M

Slither static analysis findings on `PqcInvoiceEscrow.sol`

1.10M gas

Dry-run deploy verified on live RPC `0xabe5` (Chain 44005)

LIVE WEB CONSOLE

`https://alphonshhhomenicmtu.github.io/aevum-pqc-escrow/`

GITHUB REPOSITORY (APACHE-2.0)

`https://github.com/alphonshhhomenicmtu/aevum-pqc-escrow`

DEMO VIDEO (YOUTUBE)

`https://www.youtube.com/watch?v=nkpk0sRM6qs`

Why Long-Lived Stablecoin Escrows Need Post-Quantum Authorization

Standard ECDSA signatures on secp256k1 face existential quantum risk, while B2B invoices still require gasless UX on Shanghai EVM.

Classical ECDSA-Only Escrow Vulnerabilities

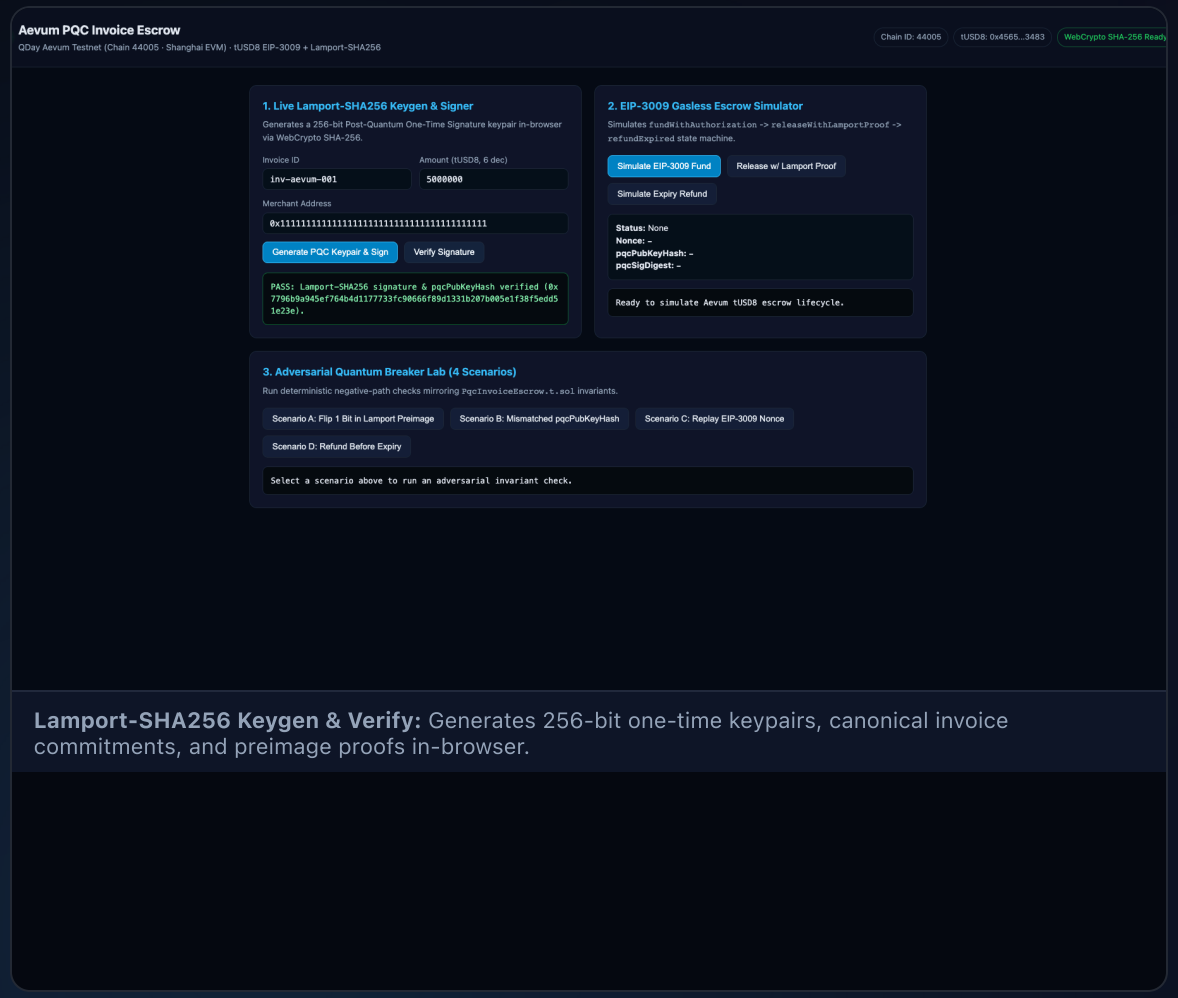
- **Shor's Algorithm Exposure:** Any exposed ECDSA public key allows a future fault-tolerant quantum computer to derive the private key and drain escrowed funds.
- **No Hash-Based Release Gate:** Standard escrows rely solely on `ecrecover`, offering zero quantum-resistant proof for high-value invoice releases.
- **EVM Compatibility Gaps:** Many PQC schemes require custom precompiles unavailable on Shanghai-era EVM chains like QDay 44005.

Aevum PQC Escrow Design

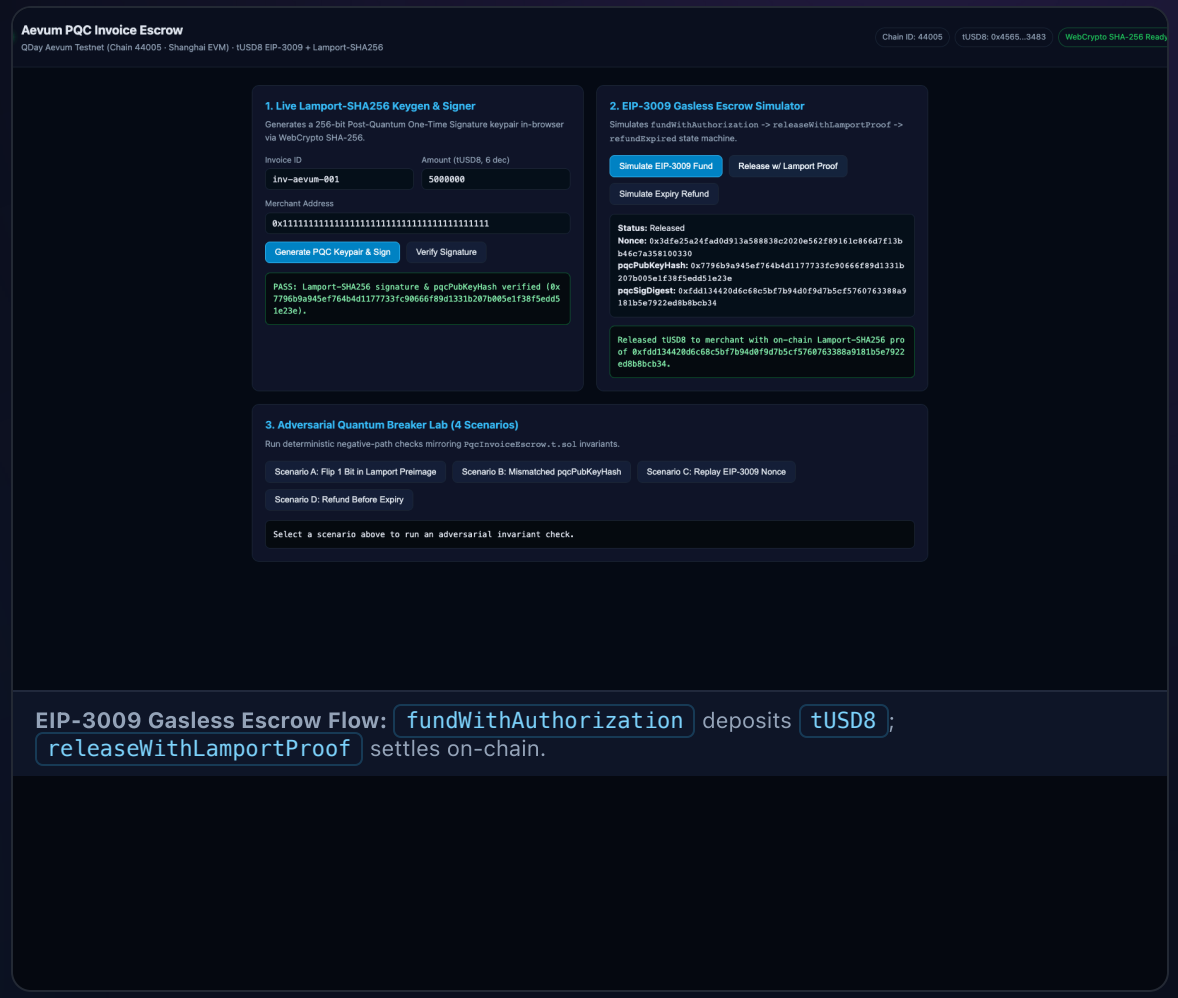
- **On-Chain 256-Bit Lamport-SHA256:** Relies purely on collision-resistant SHA-256 preimage proofs (quantum-safe under Grover's bound) on Shanghai EVM.
- **Gasless EIP-3009 Funding:** Supports both standard `fund()` and gasless `fundWithAuthorization()` on `tUSD8`.
- **One-Time Key Binding & Expiry:** Enforces single-use Lamport public-key commitments and deterministic `refundExpired()` recovery.

WebCrypto / Python PQC Signer → `PqcInvoiceEscrow.sol`

Zero-dependency Python & browser WebCrypto Lamport-SHA256 keygen/sign/verify paired with EIP-3009 gasless stablecoin funding.



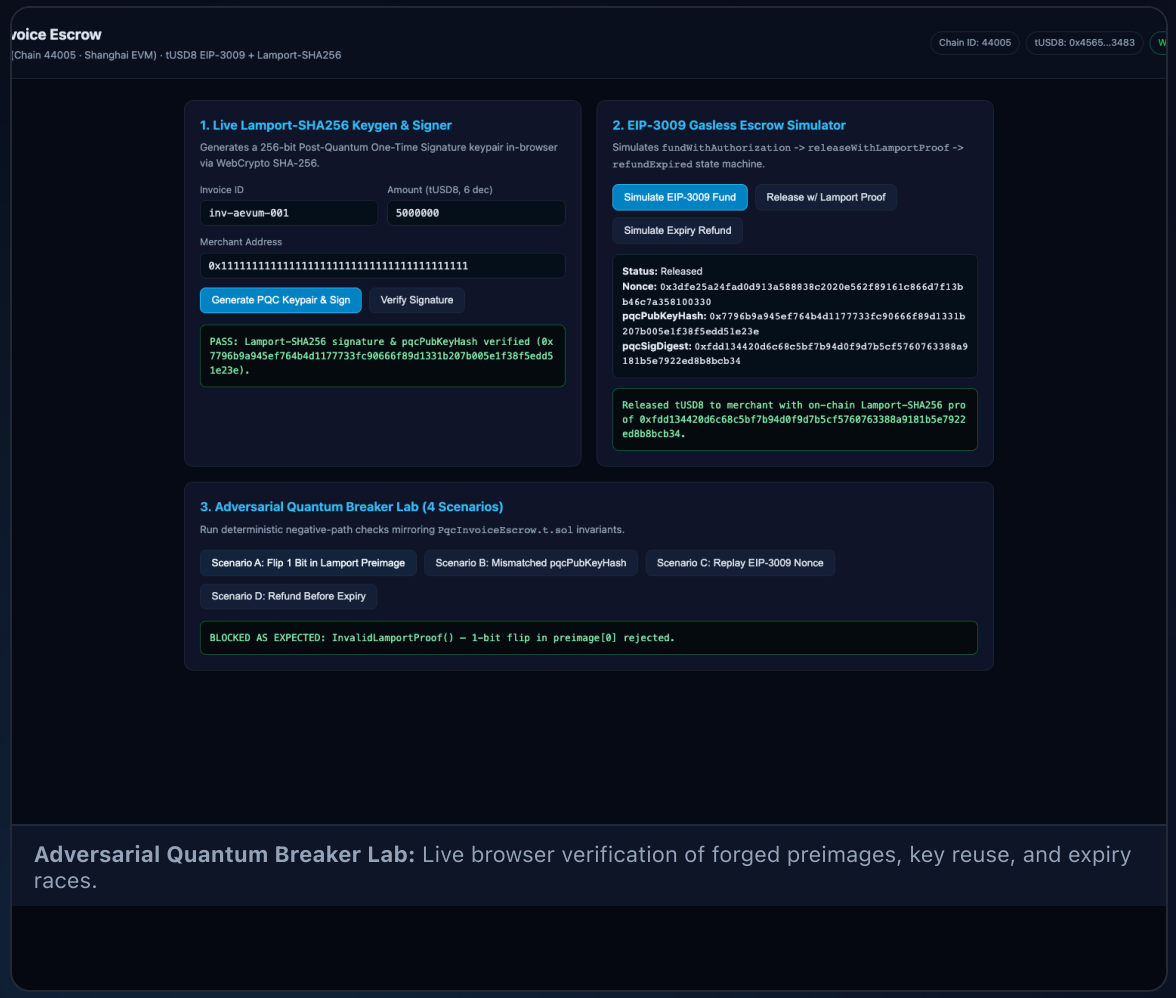
Lamport-SHA256 Keygen & Verify: Generates 256-bit one-time keypairs, canonical invoice commitments, and preimage proofs in-browser.



EIP-3009 Gasless Escrow Flow: `fundWithAuthorization` deposits `tUSD8`; `releaseWithLamportProof` settles on-chain.

Interactive Adversarial Testing in the Browser Console

Judges can trigger four deterministic attack scenarios against `PqcInvoiceEscrow.sol` logic and verify immediate rejection.



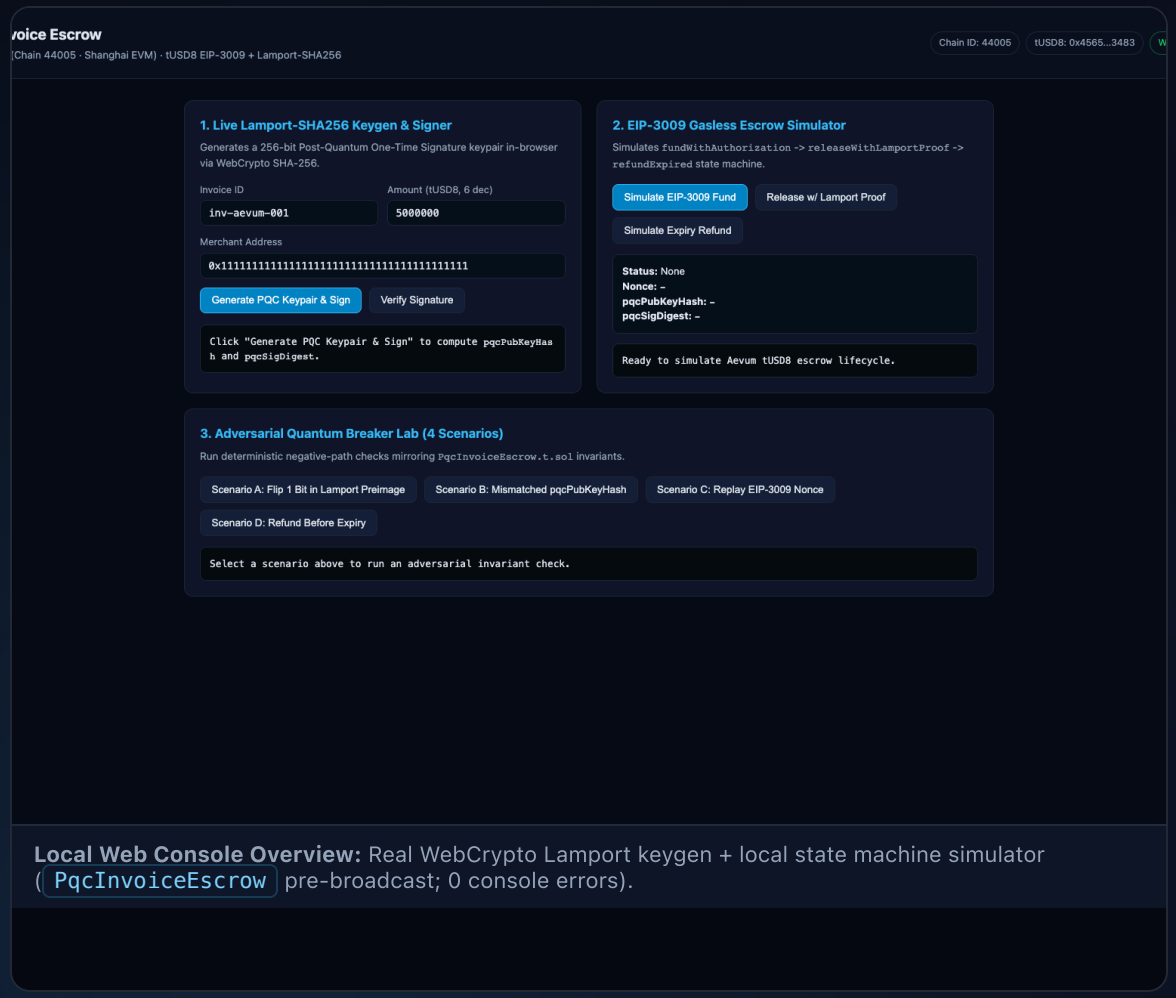
Adversarial Quantum Breaker Lab: Live browser verification of forged preimages, key reuse, and expiry races.

Four Tested Adversarial Attack Vectors

- **1. Forged Lamport Preimage:** Corrupting even a single bit-branch in the 256-element SHA-256 signature fails on-chain commitment verification.
- **2. One-Time Signature Replay:** Attempting to reuse a consumed Lamport public key or already-settled invoice ID reverts immediately.
- **3. Expired Invoice Release Race:** Calling `releaseWithLamportProof` after expiry or double-claiming after `refundExpired` is blocked.
- **4. Relayer Term Hijack & EIP-3009 Replay:** 8-word nonce (`chainid, escrow, invoiceId, payer, merchant, amount, expiresAt, pqcPubKeyHash`) rejects any relayer field swap or replay before state mutation.

Foundry 15/15 (Shanghai) + Pytest 31/31 + Slither 0 H/M + QDay Dry-Run

Verified offline with Foundry (`solc 0.8.28`, `evm_version = "shanghai"`), Python pytest, Slither, and live QDay Chain `44005` RPC.



Local Web Console Overview: Real WebCrypto Lamport keygen + local state machine simulator (`PqcInvoiceEscrow` pre-broadcast; 0 console errors).

Test & Static Analysis Proof

- **Foundry (`forge test --offline`):** `15/15` passing (14 unit/adversarial tests + 1 stateful invariant suite: 256 runs / 8,192 calls, 0 reverts).
- **Python PQC Verifier (`pytest -q`):** `31/31` passing (8-word EIP-3009 nonce binding & canonical SHA-256 invoice commitment parity).
- **Slither & Gitleaks:** `0` High / `0` Medium issues on `PqcInvoiceEscrow.sol`; `0` secrets leaked.

Live QDay Chain `44005` RPC & Deploy Dry-Run

- **RPC Endpoint:** `https://rpc-test.qday.info` live (`eth_chainId = 0xabe5`); `PqcInvoiceEscrow` not yet broadcast.
- **Turnkey Script:** `scripts/deploy_aevum.sh --dry-run` simulation passed (`PqcInvoiceEscrow`, `1,102,379` gas).
- **Stablecoin Target:** `tUSD8` at `0x456569838f1d43009A91ebA4263147Afd7233483`.

Production Post-Quantum Settlement Roadmap on QDay

Apache-2.0 open-source foundation ready for live QDay testnet broadcast and multi-invoice PQC key trees.

Phase 1 · Live Chain 44005 Deploy

- Broadcast `PqcInvoiceEscrow.sol` via `scripts/deploy_aevum.sh`.
- Bind live `tUSD8` EIP-3009 relayer for zero-gas payer onboarding.

Phase 2 · Merkle / XMSS Key Trees

- Upgrade from single-use Lamport keys to Merkle-authenticated XMSS trees for reusable merchant signing.
- Batch invoice settlement proofs to amortize calldata gas.

Phase 3 · Merchant PQC SDK

- Publish npm & PyPI packages wrapping WebCrypto + `pqc_verifier/`.
- Drop-in checkout widget for QDay ecosystem dApps.

LIVE WEB CONSOLE

`https://alphonshhhomenicmtu.github.io/aevum-pqc-escrow/`

GITHUB REPOSITORY (APACHE-2.0)

`https://github.com/alphonshhhomenicmtu/aevum-pqc-escrow`

DEMO VIDEO

`https://www.youtube.com/watch?v=nkpk0sRM6qs`