SERENITYVAULT PROTOCOL · SERIES 8

A governance and evidence architecture for sensitive decisions

How a system can decide, prove, and account for itself — without any single authority, human or software, ever acting unchecked.

Commercial white paper · Series 8 · Current stage: ALPHA

ALPHA SerenityVault Protocol Series 8 is test-only software. It has not undergone hardware, field, legal, institutional, or regulatory validation, nor any external audit. No real keys, customer data, or vault contents ever pass through it. This document describes an architecture and its genuine state of progress — not a certified or production-deployed product.

The problem

Any system that controls access to sensitive assets eventually faces the same question: who decided what, on what basis, and how can that be proven afterward to a third party who wasn't in the room?

The usual answer rests on trust — trust in an operator, in a log assumed to be complete, or in an authority assumed incorruptible. That trust is rarely verifiable after the fact, and it becomes a single point of failure: if the operator errs, if the log is altered, or if the authority exceeds its mandate, nothing in the system itself reveals it.

SerenityVault Protocol starts from a different premise: a sensitive decision should never rest on the word of whoever made it alone. It should produce, at the moment it is made, sealed evidence, a tamper-evident record, and an explainable trail — independently verifiable later by anyone who needs it: an auditor, a regulator, a court.

The SerenityVault approach

The protocol organizes every sensitive decision into four mandatory steps, each owned by a distinct component that the others cannot bypass.

TCAC — Sealed classification

Every action is first classified against a fixed, versioned five-category schema. Any change to

ADÈLE — Decision engine

A deterministic rule engine renders an explicit decision — allow, deny, or escalate to a human

the schema produces a new version and a new hash — nothing changes silently.

— with a structured justification. When in doubt, denial is the default.

METATRON — Evidence sealing

Every piece of evidence tied to a decision is sealed with a cryptographic hash (SHA3-512), making tampering detectable.

Hash-chained journal & Merkle root

Decisions and evidence accumulate in an append-only, chained, verifiable journal, summarized by a Merkle root: altering any past entry invalidates the entire chain.

Multi-profile SAK reports

The same journal can produce reports tailored to three different readers — auditor, tribunal, regulator — with controlled redaction of sensitive information per profile.

Independent verifier

A second implementation, written in a different language (Rust) and tested to match the first exactly, lets a third party verify evidence without trusting the original implementation.

The governance principle: no single authority

The protocol distinguishes two access levels. The first (Tier 1) concerns decisions and the assets themselves. The second (Tier 2) concerns after-the-fact analysis, audit, and investigation. A Tier 2 component can never access Tier 1 directly — the separation is structural, not merely procedural.

Némésis, the protocol's forensic compilation module, illustrates this principle. It operates strictly at Tier 2, under supervision, and its function is exclusively to compile facts that have already been sealed. It never judges innocence, guilt, or truth — that responsibility remains human at every step.

Where the protocol stands today

The software roadmap planned for Series 8 comprised twelve steps; all twelve have been delivered within ALPHA scope: initial repository, TCAC classification, ADÈLE engine, METATRON evidence format, chained journal, Merkle root and SAK report, independent Rust verifier, sandboxed API, explainability and contradiction scanning, multi-profile SAK v2, Némésis specification and supervised compilation, and a reproducible end-to-end demonstration.

This software milestone does not equate to a production deployment. Here is precisely what has not yet been done:

- No hardware, field, legal, institutional, or regulatory validation has been conducted.
- No external security audit has been performed.
- The system uses no production cryptography and no real keys; signatures remain test placeholders.
- No real customer data or vault contents have ever passed through the system.
- The API and components exist in a local sandbox environment — they are not deployed in production.

A commitment on language

The protocol commits to never using the terms `irrefutable`, `absolute proof`, `unhackable`, or `Production-Proven` to describe its current state. Every output of the system must explicitly distinguish its maturity stage — ALPHA, Beta, Release Candidate, or Production-Proven — and never present an earlier stage as equivalent to a later one. We favor an accurate description over an impressive one.

What comes next

The protocol's next maturity stages — Beta, Release Candidate, and eventually, where warranted, the Production-Proven designation — depend on validations that have not yet taken place: hardware and field testing, independent security audits, and, depending on the use context, legal, institutional, or regulatory review. None of these stages is presumed achieved by this document.

Discuss a pilot or a partnership

For questions about the architecture, the roadmap, or to consider a supervised evaluation of the protocol in its current ALPHA state, contact the SerenityVault team.

`info@serenityvault.com · protocole.com`

SerenityVault Protocol, Series 8 — Commercial white paper. A descriptive document about a software architecture at ALPHA stage; it does not constitute a certification, a security guarantee, or legal or financial advice.