Cannot Certify, So We Encode the Obligation: A Collaborative-zk-SNARK Re-Audit as a Witness-Validation Negative Result for Federated SPARQL

Jesse Wright · the sparq project

A negative-result / security-engineering contribution. It is not *a security, soundness, privacy, or attestation proof: the collaborative (multi-prover) proof path it audits is* ***unbuilt*** *— every method on the boundary returns a fail-closed* `NotYetImplemented` *— so the re-audit* ***cannot certify soundness*** *and makes no such claim. The contribution is the adversarial re-audit methodology, the honest finding that the path is* ***not yet*** *sound to claim, and a* ***witness-validation-before-proving*** *test obligation (R-WV) that converts a cryptographic precondition from a documented caveat into an enforceable build-time gate. The estate is under two open audit gates — an external single-prover cryptographer audit (*`sq-qhy4`*) and this collaborative re-audit (*`sq-9hrn`*) — neither of which is discharged here.*

Abstract

A federated query engine that wants to let several data holders jointly prove one statement over their private graphs is reaching for a *collaborative* zero-knowledge SNARK: an MPC among the provers that produces a single proof a relying party can check with an ordinary single-prover verifier. CRYPTO'25 (eprint 2025/1026) showed that this template carries failure modes that are easy to miss — most sharply, that proving over an *inconsistent or maliciously extended* witness can leak an honest prover's private inputs even when the verifier still rejects the proof, and that the folklore "honest-majority semi-honest gives malicious security for free" result holds for collaborative proving *only if the extended witness is validated before proving*. We report an adversarial re-audit of one engine's *intended* collaborative path against these failure modes. The honest disposition is a *negative result*: the path is **unbuilt** — 6 proof/attestation entry points return a gate-naming `NotYetImplemented` rather than executing — so there is no prover to forge against and **no soundness can be certified**. We assign each of the 4 2025/1026 lenses a *RE-OPEN* verdict (never CLOSED), state plainly that the validate-before-prove precondition is *unmet*, and that the closest usable third-party stack is itself *unaudited*. The durable output is *R-WV*, a 5-clause witness-validation test obligation that we encode as a build-time gate: a passing meta-test pins the current fail-closed posture and fires if a future prover ever returns success while the obligation is unmet, and an `#[ignore]`d obligation suite is written against the contract the eventual prover must satisfy. We make no security, privacy, or attestation claim; we make the gap *enforceable* so it cannot silently re-open when the path is built.

1. What this paper is, and is not

We are explicit before any technical content, because the framing is the contribution.

This paper **is**:

- an *adversarial re-audit methodology* — mapping a published set of collaborative-zk-SNARK failure modes (2025/1026) onto a concrete *intended* stack, lens by lens, and assigning each a verdict that is CLOSED only when the design *provably avoids* the pitfall and RE-OPEN otherwise;
- an *honest negative finding* — the audited path is **not yet** sound to claim, and the reasons are named, not hedged;
- an *engineering artifact* — a test obligation (R-WV) that turns the literature's precondition into a gate a build can check.

This paper **is not**, and must not be read as:

- a proof that the collaborative path is sound — it is **not yet** sound, and we **cannot certify** it (there is nothing built to certify);

- a privacy, integrity, or attestation guarantee for the engine — **none** is asserted; the estate provides *no* production guarantee to a relying party pending external audit;
- a claim that the honest-majority "free" malicious-security result applies — whether it applies to this engine's specific relation is an **open** question for the eventual external audit;
- a performance result — the collaborative path has *zero* timing data points and we assert no number about it.

Two open audit gates govern everything below and are never softened: the external single-prover cryptographer audit (`sq-qhy4`) is **open**, and this collaborative re-audit (`sq-9hrn`) is itself the *gating* disposition for the multi-prover path. No security property may be claimed proven until both close; this paper closes **neither**.

2. Background: the collaborative-proving template and its hazards

The intended path is the standard collaborative-zk-SNARK shape. Several holders are the provers; each holds a private witness — a committed graph, the values it discloses, the row encodings. They run an honest-majority MPC to compute a federated correctness relation over secret-shared values and to jointly emit one proof, checkable by an *unchanged* single-prover verifier. This is exactly the template 2025/1026 studies, and two of its results bite directly:

- **Privacy is not free.** Even where soundness would survive, a malicious prover who steers the MPC onto an inconsistent or extended witness and then induces an opening can exfiltrate honest provers' private witness bits. The defence is to *validate the extended witness for consistency before the proving phase opens or commits to anything derived from it.*
- **"Free" malicious security is conditional.** The honest-majority "semi-honest gives malicious security for free" folklore holds for collaborative proving *only if* the extended witness is validated before proving, and generic semi-honest-to-malicious MPC compilers are *not* safe to apply naively to the proving setting.

The engine's MPC layer is honest-majority by construction, which places it in the regime where the "free" result *could* apply — and that is precisely what makes the validate-before-prove precondition the whole game, because the antecedent is satisfied and only the consequent is in doubt.

3. The re-audit: four lenses, every verdict RE-OPEN

We audited the intended path against 4 lenses, each a documented 2025/1026 failure mode. A lens earns CLOSED only if the design *provably avoids* the pitfall; otherwise it is RE-OPEN with the specific unmet precondition named. *Every* lens is RE-OPEN — not because a present exploitable hole was found (there is no shippable prover to forge against), but because the path is unbuilt and the preconditions are unfilled.

Lens	2025/1026 failure mode	Verdict
1	Witness-extension leakage — proving over an inconsistent / extended witness can leak honest inputs even when the verifier rejects.	RE-OPEN
2	Malicious-compiler insecurity — a generic semi-honest-to-malicious compiler is *not* safe to apply as-is to a collaborative prover.	RE-OPEN
3	Honest-majority semi-honest attains malicious security *iff* the extended witness is validated before proving.	RE-OPEN
4	The closest usable third-party collaborative-proving stack is *unaudited* and predates the paper.	RE-OPEN

Table 1: The 4 re-audit lenses and their verdicts. *Every* verdict is RE-OPEN: the trust model places the engine in the regime where the honest-majority "free" result *could* apply, but the validate-before-prove precondition is unmet and unencoded, no collaborative-proving construction is selected, and the nearest usable dependency is unaudited. A RE-OPEN is *not* a finding of a live exploit — it is the honest disposition for an unbuilt, audit-gated path, and a CLOSED verdict on any lens would be premature.

evidence: `research/mpc-cozk-reaudit.md §2 per-lens table + the four '### Lens N' sections (bead sq-9hrn); each maps a documented eprint 2025/1026 collaborative-zk-SNARK failure mode onto the intended multi-prover stack` (environment: canonical)

Lens 1 (witness-extension leakage) is the live shape. The MPC layer already documents the same leak at its own level: a mid-pipeline open at the minimal honest-majority threshold carries no redundancy to catch an inconsistent share, and opening a value computed on an inconsistent witness is — per 2025/1026 — a confidentiality hazard, not only a correctness one. The planned mitigation (authenticate every shared value and batch-check before any open) is the MPC-layer analogue of the paper's validate-before-prove defence; it is correct in intent and *not yet landed*. For the collaborative-*proving* layer specifically, the validate-the-extended-witness-before-proving check had *no* design artifact and *no* test obligation before this re-audit. Lens 3 is the load-bearing one: the honest-majority antecedent is satisfied by construction, so the consequent — validate before proving — is the single most important gating requirement, and it is *unmet* because no proving phase exists. Lenses 2 and 4 are dependency-and-construction risks: no collaborative-proving construction has been chosen and patched, and the nearest usable stack is explicitly *unaudited* and predates the paper, so adopting it would import an *unverified* cryptographic trusted-computing-base.

4. The negative result is grounded in an unbuilt path

The honesty of this contribution rests on a checkable fact: there is nothing to certify because the collaborative path is *not built*. On the proof boundary, 6 entry points — the joint-proving `prove` and `verify`, the distributed `attest_source`, and their stub-test counterparts — each return a gate-naming `NotYetImplemented` rather than executing, and the in-circuit distributed signature over the secret-shared witness is the deferred spike the project itself calls "the join nobody has built."

Committed structural fact	Count
Collaborative-proof / attestation entry points returning a gate-naming `NotYetImplemented`	6
2025/1026 failure-mode lenses, *all* assigned RE-OPEN (none CLOSED)	4
Confirmed findings in the prior *single-prover* verifier audit this path would build on	12

Table 2: The committed structural facts behind the negative result — deterministic source/document scans over committed code and the re-audit record, *not* measurements. The 6 fail-closed entry points are the evidence that there is no prover to forge against; the prior single-prover audit's 12 confirmed findings are cited only to show the foundation is itself under an open external-audit gate (`sq-qhy4`), not to claim either is fixed.

evidence: `crates/sparq-mpc/src/proof.rs` — the `CollaborativeProof::{prove, verify}`, `Attestation::attest_source` trait methods plus their stub-test counterparts, each returning `MpcError::NotYetImplemented` with a named gate; the deferred-prover meta-test `gate_state::deferred_prover_never_proves_over_any_witness` in `witness_validation_tests.rs` pins the `fail-closed` posture (environment: canonical)

This is why the re-audit *cannot certify soundness*, and says so. There is no collaborative prover to attack; the verdicts are about whether the *intended design* avoids documented failure modes, and the answer is "not yet — the preconditions are unfilled." The current honest posture — every collaborative-proof method failing closed with a gate-naming error — is the *correct* state for an unbuilt, audit-gated path; the re-audit's job is not to bless it but to make the gate enforceable when the path is eventually built.

The foundation matters too. The single-prover verifier the collaborative path would plug into is itself under an open external-audit gate: a prior adversarial audit recorded 12 confirmed findings on a v1 verifier (a v1 documented *not* sound), and while the estate was internally re-audited as sound-as-landed for its stated threat model, no accredited external cryptographer has reviewed it (`sq-qhy4`, *open*). Critically, `sq-qhy4` audits the *single-prover* verifier only — it does *not* discharge the multi-prover construction this re-audit governs. So the collaborative path inherits an *unverified* foundation on top of an *unbuilt* superstructure: two independent reasons no soundness claim is available.

5. R-WV: encoding the precondition as an enforceable gate

The durable output is to stop the validate-before-prove precondition from being a documented caveat that a future implementer could miss, and make it a gate a build *checks*. We state it as a requirement and decompose it into a 5-clause test obligation.

Requirement (R-WV). *A collaborative-proving implementation must validate the shared extended witness for cross-holder consistency, and must abort fail-closed before any value derived from the extended witness is opened or committed into the joint proof, whenever the extended witness is inconsistent or maliciously extended.* No "prove-anyway-and-let-the-verifier-reject" path may exist — that path *is* the 2025/1026 leak.

Clause	The obligation it encodes
T1	Inconsistent-share *abort before open*: an off-codeword share of a witness value must drive a fail-closed abort *before* any open or proof-commit step runs — asserted via an instrumented round-counter showing zero witness-derived opens after the inconsistency is introduced, and *no* proof emitted.
T2	Witness-extension *leakage probe*: a witness inconsistent with its signed commitment must abort before proving, and an honest holder's private bits must be information-theoretically *unrecoverable* from the transcript up to the abort (zero opens on honest-derived lineage).
T3	Validate-before-prove is *load-bearing, not advisory*: a differential test that, with validation disabled, the same adversarial input would otherwise reach an open/proof step — pinning the gate onto the critical path so a refactor cannot silently move proving ahead of validation.
T4	*Commitment-binding* of the validated witness: a prover that validates witness A but proves over witness B must be rejected at the binding seam — closing the bait-and-switch between the validated and the proven witness.
C	Construction-*provenance* assertion (documentary / CI, not a runtime test): the adopted construction must record the specific 2025/1026-patched variant it instantiates (*not* a naive semi-honest-to-malicious compiler), and if a third-party stack is used, its pinned version plus a re-run of this lens-set against that exact version.

Table 3: The 5-clause R-WV test obligation (re-audit §3). T1–T4 are runtime obligations the eventual prover must satisfy; clause C is a documentary / CI provenance assertion. The obligation is *necessary* and explicitly *not* claimed *sufficient* — it encodes the 2025/1026 precondition and the bind-the-validated-witness discipline; full soundness additionally requires the external multi-prover audit, which is *not* in scope here.

evidence: `research/mpc-cozk-reaudit.md §3 (the R-WV requirement) — T1, T2, T3, T4 + clause C; encoded in crates/sparq-mpc/src/witness_validation_tests.rs (bead sq-7leq)` (environment: canonical)

The encoding is two-tier, and its disposition is *OPEN, not met*. First, a *passing* meta-test pins the current fail-closed posture: `prove` refuses with a gate-naming error and never opens a witness-derived value or emits a proof, so no prove-over-an-invalid-witness path exists today — vacuously, because no prover exists. That meta-test is the regression anchor: it fires if a future `prove` ever begins returning success while R-WV is still unmet. Second, the T1–T4 suite is `#[ignore]`d — a documented *open* obligation, each clause citing the re-audit and the audit gate — written against the prover contract the eventual implementation must satisfy, to be un-ignored when that prover lands. This makes *no* soundness claim and closes *no* lens; it converts the precondition from a documentary caveat into a build-time-measurable gate, so the gap cannot silently re-open. Clause C remains a documentary obligation, because no construction has been adopted to record.

6. What this re-audit cannot conclude

We state the limits as plainly as the result, because an adversarial audit of an *unbuilt* path is honest only if it does.

- **It cannot certify the path is sound.** There is no collaborative prover to forge against; the verdicts concern whether the *intended design* avoids documented failure modes, and the answer is "not yet — preconditions unfilled." A CLOSED verdict on any lens would be premature and is deliberately withheld.
- **The "free malicious security" applicability is unproven here.** The 2025/1026 positive result is stated for a *class* of constructions; whether this engine's eventual federated-correctness + commit-

ment-fold relation, under its chosen sharing scheme and proof system, lands inside that class is an *open* question for the eventual external audit.
- **The third-party-stack risk is qualitative.** Absent an audit of a specific pinned version, we can only flag the nearest usable dependency as *unaudited*; we *cannot* say it is broken, only that it is *unverified* against the very failure modes this re-audit is about.
- **Performance is out of scope with zero data points.** The collaborative path has *no* timing number, and we assert none.

The single durable output is the R-WV requirement and its 5-clause obligation: it turns the 2025/1026 precondition from a caveat into an enforceable gate. That is the whole contribution, and it is a *negative* one.

7. Related work and honest positioning

Collaborative zk-SNARKs (the line eprint 2025/1026 extends), honest-majority MPC, and IT-MAC authentication are established cryptographic work, and we claim no novelty in any of them. We also claim no novelty in the *content* of the 2025/1026 results; we *apply* them. The contribution is the *methodology and the artifact*: an adversarial re-audit that maps a specific published set of collaborative-proving hazards onto a concrete intended stack and refuses a CLOSED verdict the evidence does not support, plus the R-WV test obligation that encodes the load-bearing precondition as a build-time gate against an *unbuilt* path. This is a security-engineering *lessons* result, not a cryptographic theorem — and its honesty is the point: it reports that the path is *not yet* sound to claim, names every reason, and ships a gate rather than a guarantee. The genre — a published *negative result* that overturns the comfortable assumption that an honest-majority semi-honest collaborative prover is "free" — is itself the kind of contribution the literature asks for and the project's empirical-honesty mandate requires.

8. Conclusion

A federated engine that wants several holders to jointly prove one statement over their private graphs is reaching for a collaborative zk-SNARK, and CRYPTO'25 showed that template hides leakage and conditional-security pitfalls that are easy to miss. We re-audited one engine's *intended* path against 4 of those pitfalls and reached an honest *negative result*: the path is **unbuilt** — 6 proof/attestation entry points fail closed with a gate-naming error — so we **cannot certify soundness** and assign *every* lens RE-OPEN. The foundation it would build on is itself under an open external audit (`sq-qhy4`, 12 prior single-prover findings), and this collaborative re-audit (`sq-9hrn`) is the gating disposition for the multi-prover path. We claim *no* security, privacy, or attestation property — there is nothing built to claim it of. What we contribute is the methodology, the honest finding, and R-WV: a 5-clause witness-validation-before-proving obligation, encoded as a build-time gate that pins the current fail-closed posture and fires if a future prover ever proves while the precondition is unmet. The value is not a guarantee — it is that the gap is made *enforceable* instead of merely documented, so it cannot silently re-open when the path is finally built.

---

sparq project · this paper is a NEGATIVE-RESULT / security-engineering-lessons contribution and asserts *no* proven security, privacy, soundness, or attestation property. Evidence traces to the adversarial re-audit `research/mpc-cozk-reaudit.md` (bead `sq-9hrn`), the encoded test obligation `crates/sparq-mpc/src/witness_validation_tests.rs` (bead `sq-7leq`), the deferred proof boundary `crates/sparq-mpc/src/proof.rs`, and the prior single-prover audit `research/zk-soundness-audit.md` under the open external-audit gate `sq-qhy4`. The estate is research-grade and *not* externally audited; the collaborative path is unbuilt. Counts in this document are injected at build time from the paper-bound evidence file; see the provenance stamp on the published page.