# Zero-Knowledge Proof of Correct SPARQL Evaluation over Verifiable Credentials

Jesse Wright[1][0000−0002−5771−988X], Sir Nigel Shadbolt[1], Jun Zhao[1], Rui Zhao[3], and Christoph Braun[2]

[1] University of Oxford, Department of Computer Science, Wolfson Building, Parks Road, Oxford OX1 3QG, UK
jesse.wright@cs.ox.ac.uk
[2] Karlsruhe Institute of Technology (KIT), Institute AIFB, Karlsruhe, Germany
[3] Tianjin University, Tianjin, China
https://me.ryey.icu/

**Abstract.** Verifiable Credentials let an issuer sign attribute claims about a holder, and selective-disclosure schemes let the holder reveal subsets of those claims; but no peer-reviewed system lets the holder prove that a SPARQL query — any non-trivial algebra over multiple credentials — was evaluated correctly against the holder's wallet without revealing more than the query result entails. We close that gap with a circuit-level encoding of a non-trivial fragment of the SPARQL 1.1 algebra (BGP, Join, Filter, OPTIONAL, UNION, bounded property paths, EXISTS, NOT EXISTS, and MINUS) over Merkle-committed signed RDF, modular across both commitment shape (sorted-leaf Merkle and prefix tree) and signature suite (Schnorr, BBS+, SD-JWT-VC, ed25519, ECDSA, with ML-DSA / lattice-BBS as the post-quantum migration target). The encoding is realised as a Noir circuit library; its soundness and completeness are stated against the Pérez–Arenas–Gutiérrez algebra and mechanised in Lean 4 over a Lampe-extracted Noir operational semantics, with a worked numeric primitive (IEEE 754 `add_f32`) closed end-to-end and the SPARQL-fragment proofs reported with explicitly enumerated open obligations. A *disclose-and-verify-externally* discipline keeps any deterministic function of the disclosed multiset (DISTINCT, ORDER BY, LIMIT, COUNT, SUM) outside the circuit, so the prover pays no constraints for properties the verifier can re-check on the released bindings. We report empirical results for the sparql_noir implementation against the W3C SPARQL 1.1 evaluation suite and the new zkSPARQL-bench axial benchmark over partitioned LUBM and BSBM, qualifying the prior zkVM baseline's sub-second-prover commitment for queries over up to $10^3$ committed triples. A draft W3C-style specification for SPARQL-as-default-VC-query-interface ships as a companion artefact.

**Keywords:** Zero-knowledge proofs · SPARQL · Verifiable Credentials · RDF · Noir · Formal verification.

## 1 Introduction

Verifiable Credentials let an issuer sign attribute claims about a holder; existing selective-disclosure schemes let the holder reveal subsets of those claims. No peer-reviewed system, however, lets the holder prove that a SPARQL query — any non-trivial algebra over multiple credentials — was evaluated correctly against the holder's wallet without revealing more than the query result entails. This paper closes that gap.

*Motivation.* Consider four friends applying jointly for a rental. Each holds a Solid Pod [38] containing W3C Verifiable Credentials [42] including signed payslips. The landlord requires proof that their cumulative annual salary exceeds £100,000. Under current standards, the privacy-respecting options are to hand over every payslip in full, or to ask each issuer to pre-sign per-applicant threshold flags — which still leaks salary bands and requires bespoke issuer cooperation for every predicate. The EU Digital Identity Wallet rollout (Member-State provision in 2026, private-sector acceptance in 2027 under EU 2024/1183 [12] and the ARF [11]) will issue credentials at population scale, but the issuer-must-pre-sign-every-predicate gap (e.g. `is_over_18` in ISO mDL [22]) is structural. The holder needs to evaluate arbitrary queries — here, a sum across four signers with an inequality test — and prove the result, not the inputs.

*Why SPARQL.* Three properties pick SPARQL out among candidate query languages. It is built on globally-unique IRIs [9] so entities are referenced consistently across credentials from independent issuers, unlike Cypher's local identifiers [14]. RDF and SPARQL operate under the Open World Assumption [10], the sound semantics for selective disclosure. And SPARQL admits a compositional, endpoint-independent algebra [33], unlike GraphQL [43]. The closest credential-query language, OpenID4VP's DCQL [24], supports *selection* across credentials with field equality and set logic; we provide *evaluation*.

*Threat model.* The holder is the prover and may be malicious; the verifier trusts only issuer signatures and the proof; issuers are trusted to sign honestly; the network is passive. Full statement in §7.

*Trajectory.* The CEUR Vol-4085 predecessor of this work [47] demonstrated feasibility for SPARQL 1.1 `SELECT` inside the RISC Zero [36] zkVM at roughly seven and a half minutes for a trivial query over twenty-three triples — a feasibility milestone committing the follow-up to a circuit-DSL approach. This paper is that follow-up. We supersede the zkVM baseline with a Noir [3] circuit-level evaluator over a non-trivial SPARQL 1.1 fragment, mechanise soundness in Lean 4 via Lampe-extracted Noir semantics [35], and ship a draft W3C-style specification. Federated-MPC and authentication-as-query research questions are out of scope (§9).

*Contributions.*

1. **A circuit-level encoding of a non-trivial SPARQL 1.1 fragment** — BGP, Join, Filter, OPTIONAL, UNION, property paths, EXISTS, NOT EXISTS, MINUS — over Merkle-committed signed RDF, modular across commitment shape (sorted-leaf Merkle, prefix tree) and signature suite (Schnorr, BBS+, SD-JWT-VC, ed25519, ECDSA; ML-DSA / lattice-BBS as PQ migration target). (§4, §5.)
2. **A soundness and completeness statement** for the encoding, with a layered Lean 4 + Lampe + `proven-zk` strategy [35]. A worked numeric primitive (IEEE 754 `add_f32`) is closed end-to-end; SPARQL-algebra obligations are mapped honestly with named `sorry`s (§6).
3. **The disclose-and-verify-externally principle**: information in the disclosed output must *not* be re-proven in-circuit; any deterministic function of the disclosed multiset (DISTINCT, ORDER BY, LIMIT, COUNT, SUM, AVG, MIN, MAX) is checked externally. We pay no constraints that Crescent [31] and ZKSQL [23] do. (§5.)
4. **A modular comparison study** across commitment shapes and signature suites on zkSPARQL-bench (LUBM [17], BSBM [6]), reported axially with PoneglyphDB [16] and the zkVM baseline; we verify or qualify the sub-second prover commitment of the predecessor. (§8.)
5. **A draft W3C-style specification** for SPARQL-over-VC, the lever for ecosystem adoption via the W3C Credentials Community Group. (§9; `paper/spec/`.)

*Related work in brief.* Three clusters frame the contribution (full survey §3). *Signature-level selective disclosure* — BBS+ [5,26], SD-JWT-VC [13], AnonCreds [21], Crescent [31], zk-creds [37] — proves subsets of pre-signed messages but supports no algebra. *Zero-knowledge SQL* (vSQL [53], ZKSQL [23], PoneglyphDB [16]) proves correct execution over unsigned outsourced data; sparql_noir is the SPARQL/RDF analogue. *Zero-knowledge RDF/SPARQL* — our co-author's BBS+-on-RDF foundation [7] (BGP-shape, the substrate of our full algebra), Yamamoto et al.'s slide-only prototype [49,48] (BGP+FILTER; team pivoted away from SPARQL in [52]), and the zkVM baseline [47].

*Roadmap.* §2 fixes notation; §3 surveys; §4 the commitment; §5 the per-operator translation, EXISTS / NOT EXISTS, and the disclose-externally principle; §6 the soundness theorem and Lean mechanisation; §7 forgeability versus retro-disclosure; §8 implementation and benchmarks; §9 limitations and the spec artefact; §10 concludes.

## 2 Background

### 2.1 Verifiable Credentials

A Verifiable Credential (VC) is a set of attribute claims about a *holder* signed by an *issuer* and shown to a *verifier* [42]. We treat the binding between credential graph and signature as an abstract suite $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Disclose})$.

Relevant instantiations: BBS+ [5] (PoK-based unlinkable disclosure), SD-JWT-VC [13] (hash-salt over JWT), ed25519 / ECDSA Data Integrity [41], ISO mDL [22] (MSO over CBOR), and the ML-DSA / lattice-BBS migration target [27,30]. Per-suite implications: §7.

### 2.2 SPARQL 1.1 algebra

We follow Pérez–Arenas–Gutiérrez semantics [33] for the SPARQL 1.1 algebra [19]. A query is a tree of triple patterns under BGP, Join, LeftJoin, Union, $\mathsf{Filter}_{\varphi}$, $\mathsf{Project}_{V}$, Minus, and bounded property paths; SELECT modifiers (Distinct, OrderBy, Slice) and aggregates act on multisets of solution mappings. Evaluation $[\![Q]\!]_D$ is inductive from $[\![(s,p,o)]\!]_D = \{\mu \mid \mu(s,p,o) \in D\}$; this is the semantics our circuits must match (§5).

### 2.3 Zero-knowledge SNARKs and Noir

We compile circuits in *Noir* [3], Aztec's Rust-like ZK DSL targeting the UltraPLONK backend [15] over the BN254 scalar field, with soundness, zero-knowledge and argument-of-knowledge in their standard cryptographic-game forms (formalised in §6 and §7).[4] Commitments are over BN254 with Pedersen hashing and sorted-leaf Merkle trees [3], both first-class primitives in the Noir standard library.

## 3 Related Work

We position against three clusters: signature-level ZK over VCs, ZK-SQL, and ZK-RDF / ZK-SPARQL. The (S)MPC line on federated graph query is a different threat model and deferred to §9.

### 3.1 Signature-level ZK over VCs

*Selective-disclosure* schemes reveal a subset of issuer-signed messages and prove the residual signature valid. BBS+ [26,5] delivers unlinkable subset disclosure via pairing-PoK; SD-JWT-VC [13] achieves omission-based JWT disclosure (linkable, no in-circuit work); ISO mDL [22] ships the same shape over CBOR; AnonCreds [21] predates these with CL signatures and fixed-schema range predicates; zk-creds [37] wraps Groth16 over arbitrary signed payloads. Production

[4] Three properties carry Noir over zkVMs (RISC0, SP1) and lower-level alternatives (circom, Halo2, Cairo) for our workload: algebra-level circuit cost (PoneglyphDB [16] and ZKSQL [23] show circuit-level zk-SQL is 1–2 orders of magnitude cheaper than the analogous zkVM encoding, driving our move from the CEUR Vol-4085 RISC0 baseline [47] to a circuit DSL); backend-agnosticism via ACIR; and a formal-methods path through the Lampe [35] Lean 4 extraction of Noir's operational semantics, complemented by underconstrained-witness checking (`nargo check`) and SMT-level NAVe [2] (full toolchain in §6).

ZK passport circuits [40,54] are the deployed-system axis. Crescent [31] wraps a JWT or mDL in Groth16/Spartan with range proofs and validity windows — the strongest shipping multi-attribute envelope, yet still single-credential with no algebra. None of this cluster evaluates a query algebra over the credential graph.

### 3.2 Zero-knowledge SQL

vSQL [53] discharges SQL via verifiable polynomial delegation; ZKSQL [23] replaces the monolithic SNARK with per-operator authenticated set operations (two-orders-of-magnitude over naive SNARK on TPC-H); PoneglyphDB [16] compiles each SQL operator to a Halo2 circuit and composes recursively — the per-algebra-operator architecture we adopt for SPARQL: **sparql_noir is the SPARQL / RDF analogue of PoneglyphDB**. All three target an outsourced-server, closed-world model where the verifier never sees rows; we target holder-as-prover over open-world signed RDF, which flips the in-circuit cost calculus for projections and aggregates (§5).

### 3.3 ZK-RDF and ZK-SPARQL

*Braun & Käfer* [7] establish the BBS+-on-RDF foundation: an RDF-term-level ZKP encoding combining BBS+ PoK, LegoGroth16 range proofs, and a universal accumulator, sub-second on BGP. Braun is a co-author here; the algebra layer (Join, LeftJoin, Union, Filter, paths, EXISTS, Minus) plus mechanised soundness is our contribution above their primitives. *Yamamoto et al.* [48] formalised linked-data selective disclosure with a zk-SPARQL prototype for BGP-with-FILTER over termwise BBS+, documented only via slide decks [49,50] and the Veanpods / rdf-proofs codebase [51,52]; the team has since pivoted away from SPARQL.

The first author's CEUR Vol-4085 predecessor [47] compiled Oxigraph to RISC-V inside the RISC0 zkVM over Ed25519-signed credentials, supporting all Oxigraph SELECT queries but at ∼7.5 minutes for `SELECT * WHERE {?s ?p ?o}` over 23 triples on an M1 16 GB. CEUR Vol-4085 is doctoral-consortium and not independently peer-reviewed; we treat it as a thesis-trajectory artefact and benchmark directly against it (§8). The present paper is the **first peer-reviewed circuit-level ZK SPARQL evaluator over signed VCs**, with a layered Lean 4 mechanisation.

*DCQL.* OpenID4VP's DCQL [24] provides per-credential field selection, equality/prefix matchers, and boolean credential-set composition — credential-*selection* strictly weaker than our credential-*evaluation* target (§1).

### 3.4 Position summary

Table 1 positions sparql_noir against the closest neighbours along the axes used throughout the paper.

Table 1: Closing comparison: per-system feature axes with section pointers to where sparql_noir's contribution on each axis is unpacked.

| System | Algebra | Soundness | Sub-second | Section pointer |
|---|---|---|---|---|
| Braun-Käfer 2025 [7] | BGP only | Informal | Yes | §5, §6 |
| Yamamoto et al. 2022 [48] | BGP + FILTER (slides) | None | Not benchmarked | §5, §8 |
| Wright 2025 CEUR [47] | Full SPARQL 1.1 (zkVM) | zkVM-substrate | No (∼7.5 min) | §8 |
| **sparql_noir (this paper)** | SPARQL 1.1 fragment + revealed agg | Lean 4 + Lampe + `proven-zk` | Targeted; pending bench | §5, §6, §8 |

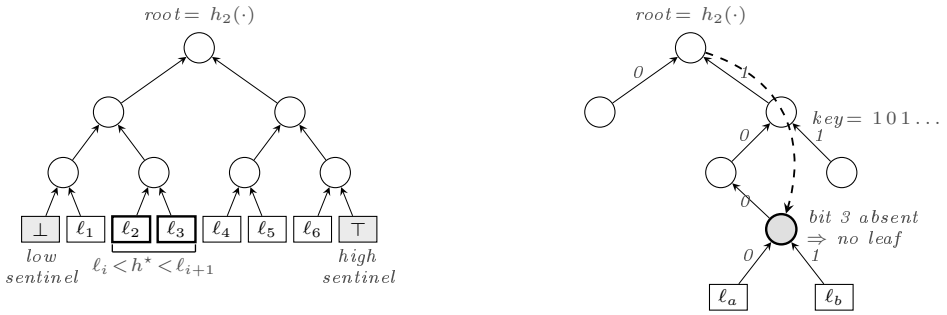
## 4 Commitment design

The commitment layer fixes what an issuer signs and what a circuit opens. The encoding of §5 dispatches on the shape from the metadata, so a single circuit family supports several schemes.

*Encoding to leaves.* Each RDF term encodes to one BN254 field element via Pedersen hash of a (type-code, value) tuple: type codes are named-nodes (0), blank-nodes (1), literals (2), variables (3), default graph (4). Strings pass through Blake3; literals carry (lexical, numeric-slot, language tag, datatype IRI) with the numeric slot filled by parsed integer / Boolean / ms-Unix timestamp for the relevant `xsd:` types and $\mathsf{Enc}_s(lex)$ otherwise. A quad encodes as $\mathsf{hash4}(\mathsf{Enc}_t(s), \mathsf{Enc}_t(p), \mathsf{Enc}_t(o), \mathsf{Enc}_t(g))$. Full rules in [9] and `spec/encoding.md`.

*Sorted-leaf Merkle (default).* `noir::utils::merkle` sorts leaves ascending by `hash4` (stable; permutation-invariance property-tested) and builds the tree under Pedersen `hash2` nodes. Sorting on collision-resistant hashes commits to the multiset without in-circuit canonicalisation [25]; depth is `merkle_depth` = 11 default (capacity $2^{11}$). Making the sort public turns non-membership into a two-leaf bracket proof [18,15].

*Prefix-tree (round-4).* A Merkle-Patricia commitment indexed by structured keys collapses non-membership over a class of triples to one path-divergence proof, unblocking query classes that incur an $|D|^t$ blow-up under sorted-leaf ($t$ = inner triples in the negated pattern, distinct from M): multi-triple `NOT EXISTS`, `MINUS` with non-ground inner, and the single-circuit `OPTIONAL` collapse with free unmatched positions. Wired but not shipped (`decisions/non-membership-sentinels-transform-wiring.md`); benchmarks (§8) cover sorted-leaf only. Fig. 1 contrasts the two non-membership witnesses side by side.




**(a) Sorted-leaf Merkle**
*Inclusion: $O(\log n)$ Pedersen $h_2$ steps. Non-membership: two adjacent inclusions plus strict-$<$ checks against $h^\star$; low/high sentinels handle the range boundaries.*

**(b) Prefix tree**
*Inclusion: $O(\log n + k)$, $k$ the key length. Non-membership: one path-divergence proof; sentinel-free, since absence is witnessed by the missing edge.*

Fig. 1: Sorted-leaf Merkle commitment (left) versus prefix-tree commitment (right). Both commit the same multiset of quad hashes; they differ in the shape of the non-membership proof. The sorted-leaf scheme brackets an absent hash $h^\star$ between two adjacent included leaves $\ell_i, \ell_{i+1}$ (the heavily-outlined pair), with mandatory low/high sentinels handling the range boundaries. The prefix tree witnesses non-membership by walking the queried key down to the first level where the expected child edge is absent (the highlighted divergence node), at the cost of one extra $h_2$ step per key bit.

*Sentinels.* Sorted-leaf requires mandatory low / high sentinels (sparql_noir #47): absent hashes outside the populated range cannot be witnessed by an internal pair. For $n$ leaves and a uniformly distributed absent hash, rank among $n+1$ gaps is uniform, so boundary probability is $\frac{2}{n+1}$ ($2\times10^{-3}$ at $n = 10^3$). We close it deterministically: the signer prepends $\text{LOW}_\text{H}$=0 and appends $\text{HIGH}_\text{H}$=$p-1$ ($p$ = BN254 modulus); both are included in the sorted root and inclusion-checked once per `NonExistenceConstraint`. A public per-constraint `boundary_case` tag (Lower, Middle, Upper) gates which non-membership primitive fires (§5.3); prefix-tree is sentinel-free.

### 4.1 Signature integration

The transform reads the suite name from `proof.cryptosuite` and dispatches to an in-circuit verifier; the rest of the pipeline is suite-agnostic (Table 3, cost relative to Schnorr-over-BN254).

The first four rows ship; ML-DSA / lattice-BBS is research-grade (no production Noir verifier; lattice-BBS analogues remain open [20]). Modular dispatch makes PQ migration a per-suite substitution (§7). Braun's BBS+-on-RDF [7] supplies the BBS+ contract. No suite is privileged: Schnorr is cheapest, BBS+ privacy-default, SD-JWT-VC the EUDI default [12]; §8 reports per-suite trade-offs.

Table 2: Sorted-leaf vs. prefix-tree commitments. M = `merkle_depth`; K = structured key length in `hash2` steps. Performance figures: §8.

| Axis | Sorted-leaf (default) | Prefix tree (round-4) |
|---|---|---|
| Commit time | $\mathcal{O}(n \log n)$ sort + $\mathcal{O}(n)$ hash | $\mathcal{O}(n\mathrm{K})$ trie insert |
| Inclusion gates | M Pedersen | M + K Pedersen |
| Non-membership | 2 incl. + strict-$<$ + adjacency | 1 path-divergence |
| Multi-position | $\lvert D\rvert^t$ (intractable) | 1 per shared prefix |
| Sentinels | Mandatory (§4) | Implicit |
| Update cost | Recompute root | Rebalance subtree |
| Best for | Static dataset | Dynamic wallet |
| Query unblocks | Ground-inner `NOT EXISTS`, `MINUS` | Multi-triple, full `OPTIONAL` |
| Status | Ships | Spec + wiring; unimplemented |

Table 3: Per-suite signature dispatch. Cost is relative to a Schnorr-over-BN254 baseline; final numbers in §8. "Unlink." = randomisable PoK of signature.

| Suite | Verification | Cost | Unlink. | PQ | Status |
|---|---|---|---|---|---|
| Schnorr / BN254 | scalar mul + Pedersen [15] | 1× | PoK lift | no | shipped |
| BBS+ [5,26] | pairing PoK [44] | ∼ 3× | native | no | shipped [7] |
| SD-JWT-VC [13] | ECDSA P-256 + hash unsalt | ∼ 2× | no | no | shipped |
| ed25519 / ECDSA [41] | Edwards / NIST scalar mul | ∼ 2× | no | no | shipped |
| ML-DSA [27] | lattice in-circuit | TODO | native | yes | future |

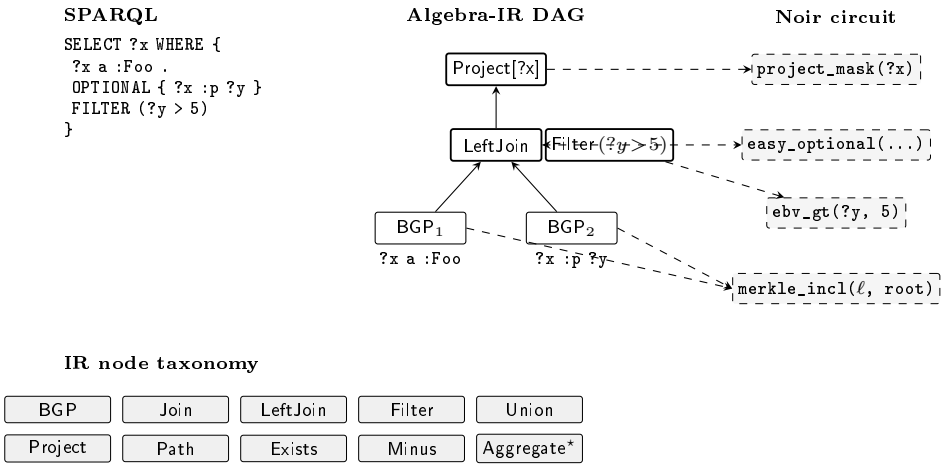
## 5 SPARQL → Noir Encoding

This section lifts the encoding of §4 into a per-operator circuit construction. The two heavyweight cases are non-membership for `NOT EXISTS` / `MINUS` (§5.3) and the tiered `OPTIONAL` collapse (§5.4); we close with the disclose-and-verify-externally principle (§5.5). Implementation lives in `sparql_noir`'s layered transform `transform/src/{parse,ir,lower,expr,emit}.rs`.

### 5.1 Algebraic intermediate representation

We translate the W3C algebra [19] under Pérez–Arenas–Gutiérrez (PAG) semantics [33] into a pure data IR (Listing 1.1; see Fig. 2 for the node taxonomy and a worked composition) before circuit emission, so lowering, expression, and emission layers can be tested and proven independently.

Lowering populates `patterns` from the BGP, threads shared variables through `bindings` / `assertions`, accumulates `filters`, and lifts UNION / OPTIONAL / NOT




*Constructors lifted from* `transform/src/ir.rs`*: each composes into* `PatternInfo` / `QueryInfo` *and emits its own per-operator Noir fragment. * Aggregate, OrderBy, and Limit are propagated to metadata and verified externally on the disclosed multiset (§5.5); they emit no in-circuit gates.*

Fig. 2: Algebra-IR shape used by the SPARQL → Noir transform. The sample query (left) lowers to the IR DAG (centre) of `Project`, `LeftJoin`, `Filter`, and two `BGP` leaves; each IR node emits an independent Noir circuit fragment (right). The full constructor palette (bottom strip) is the closed taxonomy supported by the transform; the composition pattern — one fragment per node, joined by shared binding columns — generalises to every supported query shape.

EXISTS into sub-structures. Emission walks the IR once and writes the per-query Noir program plus a `metadata.json`.

### 5.2 Per-operator translation

Table 4 fixes the per-operator recipe. The two heaviest rows (`Exists` / `Not Exists` and `LeftJoin`) are unfolded in §5.3 / §5.4.

A BGP of $k$ triples becomes $k$ `TripleInput` witnesses Merkle-checked against the signed root (§4); shared-variable equality assertions deliver Join. Filter compiles through `expr.rs` to `noir_IEEE754` [46] and `noir_XPath` with EBV from `noir/lib/ebv`; datatype hashes are constant-folded. Projection is a comptime mask emitting no constraints; EXISTS-internal variables rename `__exists_<orig>_<id>`. Union branches lower to per-branch `PatternInfo` dispatched at prove time; narrower branches pad with constraint-free slots. Property paths unfold at preprocess time: sequence / inverse / alternative / zero-or-one to BGP / Join / Union, bounded Kleene $+, *$ to a Union over depths $1 \dots k$ with $k =$ `path_segment_max` (default 8), chosen depth disclosed.

Listing 1.1: Algebra-level IR. `PatternInfo` composes BGP triples, bindings, filter expressions, UNION branches, OPTIONAL blocks, and non-existence obligations. `QueryInfo` adds projected variables and the externally-evaluated aggregate / order / slice columns (§5.5).

```
pub enum Term { Variable(String), Input(usize, usize), Static(GroundTerm) }

pub struct PatternInfo {
  patterns: Vec<ContextualizedTriple>, bindings: Vec<Binding>,
  assertions: Vec<Assertion>, filters: Vec<Expression>,
  union_branches: Option<Vec<PatternInfo>>,
  optional_blocks: Vec<OptionalBlock>,
  not_exists: Vec<NonExistenceConstraint>,
}

pub struct QueryInfo { // verifier-side fields evaluated externally
  variables: Vec<String>, pattern: PatternInfo,
  aggregates: Vec<Aggregate>, order_by: Vec<OrderKey>,
  limit: Option<usize>, offset: Option<usize>,
}
```

### 5.3 `NOT EXISTS` via sorted-commitment non-membership

Negation-as-failure over a Merkle commitment is the only new cryptographic argument outside §4; it underwrites `NOT EXISTS`, `MINUS` (W3C §18.5), and the unmatched arm of `OPTIONAL`. `FILTER(EXISTS {P})` over an outer $\mu$ holds iff some $\mu' \in [\![P]\!]_D$ is compatible; we append each inner triple to the outer BGP, hide inner-only variables, and unify shared variables through Join, reducing the `FILTER` to `assert(true)` (W3C-equivalent to $\textsc{Project}(V_{out}, \textsc{Filter}(\top, \textsc{Join}(P_{out}, P_{in})))$, no new primitive).

*`NOT EXISTS`: sorted-leaf non-membership.* At sign time `noir::utils::merkle` sorts triples ascending by `hash4` [18,15], so the issuer commits to a canonical sorted root (permutation-invariance property-tested). For `FILTER(NOT EXISTS { t })` with all positions of $t$ ground after $\mu$, the prover supplies the would-be hash $h^\star = \texttt{hash4}(t[\mu])$, two adjacent leaves $h_{left}, h_{right}$ with inclusion paths, and the index reconstruction ($right_{idx} = left_{idx}+1$). The primitive `verify_non_membership(left, right, h_star, root)` asserts $h_{left} < h^\star < h_{right}$ (strict; bag semantics is preserved without an explicit multiplicity field) plus adjacency. Direction bits in inclusion paths must be Boolean — otherwise *direction* = 2 defeats adjacency reconstruction (sparql_noir #47).

*Boundary case.* Absent hashes that sort outside the populated range cannot be witnessed by an internal pair; for $n$ leaves and a uniformly random absent hash, the boundary-slot probability is $\frac{2}{n+1}$ by order-statistic argument. Deployments add low/high sentinel triples at sign time (§4), reducing the boundary case to zero.

*Soundness reduces to:* (a) Merkle-inclusion soundness under Pedersen collision-resistance [15]; (b) the strict-ordering assertion (sorted-root canonical form);

Table 4: Per-operator translation. *Cost* = dominant gate term per output row (Pedersen-hash gates from Merkle inclusion written M = `merkle_depth`).

| Operator | PAG rule [33] | Noir primitive | Cost |
|---|---|---|---|
| BGP | $[\![(s,p,o)]\!]_D = \{\mu \mid \mu(s,p,o) \in D\}$ | Merkle incl. + position bind | M |
| Join | natural multiset join | shared-var equality | per-share |
| Filter | $\sigma_\varphi$ with EBV | noir_IEEE754 / noir_XPath + EBV | per-expr |
| Project | column projection | comptime mask | 0 |
| LeftJoin | left-join+FILTER [39] | tiered (§5.4) | bounded |
| Union | multiset union | branch bucketing | per-branch |
| Path /,\|,^,? | seq/alt/inv/zero-or-one | unfolded BGP / Join / Union | per-seg |
| Path +, * | bounded Kleene | unrolled to `path_segment_max` | $\leq k$M |
| Exists | non-empty test | flatten-to-BGP (§5.3) | M/triple |
| Not Exists / Minus | empty/diff | sorted-Merkle non-membership | 2M |

(c) the adjacency invariant (forging a non-adjacent pair reuses a sibling hash); and (d) Boolean direction bits. (a)–(b) inherit from §4; (c)–(d) are the new obligations. Lean 4 statements in §6.

*Round-3 restriction.* `NOT EXISTS` is restricted to single-triple ground-inner patterns; multi-triple, non-ground, nested, or inner-`UNION` / `OPTIONAL` cases are rejected at lowering. `MINUS` inherits via `MINUS`$\{P_o\}\{P_i\} \mapsto$ FILTER($\neg$EXISTS$\{P_i\}, P_o$).

### 5.4 Tiered `OPTIONAL` collapse

The unmatched arm of `LeftJoin` [19,39] must witness no compatible inner binding (a naive "`is_matched`=false skips inner" design is unsound under PAG §3.2). We ship a tiered easy-case predicate (preconditions: single triple pattern; positions ground after outer $\mu$; no inner FILTER; no nested OPTIONAL; depth $\leq$ `optional_cap_default` = 4). *Tier 0* (all preconditions): one `LeftJoin` circuit with `is_matched` flag, unmatched arm reusing §5.3's `NOT EXISTS`. *Tier 1* (inner FILTER): same shape with Filter row gate. *Tier N* (nested): power-set encoding, one circuit per matched / unmatched combination of $n \leq 4$ blocks, dispatched at prove time matched-count-descending; beyond 4 rejected. A pattern-shape `NOT EXISTS` under the round-4 prefix-tree (§**??**) would collapse all tiers to one circuit with $n$ flags.

### 5.5 Disclose-and-verify-externally

Our third contribution is a principle: *information revealed in the disclosed output must not be ZK-proven inside the circuit.* This excludes `DISTINCT`, `ORDER`

`BY`, `LIMIT`, `OFFSET`, `COUNT`, `SUM`, `AVG`, `MIN`, `MAX` over revealed multisets. Per-row mappings are bound to the proof via BGP inclusion, so any function of the disclosed multiset is reproducible by the verifier from public data; an in-circuit re-assertion is strictly redundant and expands the attack surface without expanding the guarantee (§6). The principle does *not* apply when a property is revealed but its underlying data stays private (e.g. "the count is 42"); there it prescribes *redesigning the disclosure* rather than adding an in-circuit primitive — this paper's fragment thus forbids `HAVING` and `GROUP BY` over hidden multisets. `EXISTS` / `NOT EXISTS` is not an exception: the boolean is the result, so §5.3 pays the in-circuit cost. Crescent [31], ZKSQL [23], and PoneglyphDB [16] pay in-circuit range proofs, aggregates, and sort-and-aggregate respectively over data they disclose; under our principle these constraints are redundant. The IR exposes `aggregates`, `order_by`, `limit` / `offset` on `QueryInfo` (Listing 1.1); emit propagates them through `metadata.json` for verifier-side execution.

## 6 Soundness

We state soundness and completeness for the encoding of §5 and report honestly on what is mechanised today. The full algebra is not closed; the methodology is demonstrated end-to-end on a worked numeric primitive (§6.3) and every open obligation sits behind a named discharge route (§6.4).

### 6.1 A three-layer methodology

*L0 — `nargo` checks.* The Noir compiler's underconstrained-witness check (`nargo check -enable-brillig-constraints-check-lookback`) [3] is default-on in CI; it catches witnesses flowing from a Brillig block without constraints but does not model arithmetic.

*L2 — NAVe SMT verification.* NAVe [2] compiles Noir to SMT-LIB and proves `verify_assert` predicates via cvc5. It is pinned to Noir beta.9 against our beta.17 (eight-beta ACIR drift); we run it against pinned snapshots in a side branch and treat it as future-work automation. NAVe verifies one circuit at a time, not circuit equivalence or match to an external reference.

*L3 — Lampe-extracted Lean 4.* For circuit-vs-reference equivalence we extract Noir to Lean via Lampe [35], lifting each definition to an `STHoare` triple. Obligations discharge against `mathlib` (arithmetic, `BitVec.Lemmas`, `bv_decide`) and `proven-zk` (Pedersen, Merkle, hash chains). Pins: Lean `v4.21.0`, mathlib `v4.21.0`, Lampe `@1cd3f4de`, `proven-zk` `@45b4de91`.

### 6.2 Soundness and completeness statements

Let $\mathcal{F}$ be the algebraic fragment of §5.2, $D$ a signed dataset committed under $(\mathcal{C}, \Sigma)$ to root $\rho$, and $\mathsf{vk}_Q$ the verification key for $Q \in \mathcal{F}$. Write $\mathsf{Eval}_{\mathrm{PAG}}(Q, D)$ for PAG semantics [33] and $\mathsf{Disclosed}(\pi)$ for the bindings revealed by a proof $\pi$ (§5.5).

**Theorem 1 (Soundness).** *For every $Q \in \mathcal{F}$, every signed dataset $D$ committed under $(\mathcal{C}, \Sigma)$ to root $\rho$ and verification key $\mathsf{vk}_Q$, and every prover-supplied proof $\pi$: if $\mathsf{Verify}(\mathsf{vk}_Q, \rho, \pi) = 1$, then $\mathsf{Disclosed}(\pi) \subseteq \mathsf{Eval}_{\mathrm{PAG}}(Q, D)$ and the underlying triples are valid against $\rho$ under $\Sigma$, except with negligible probability over the verifier's randomness and the underlying SNARK's knowledge-soundness error.*

**Theorem 2 (Completeness).** *For every $Q \in \mathcal{F}$ and every signed dataset $D$ such that $\mathsf{Eval}_{\mathrm{PAG}}(Q, D)$ is non-empty, an honest prover holding the witnesses for $D$ produces a proof $\pi$ for which $\mathsf{Verify}(\mathsf{vk}_Q, \rho, \pi) = 1$.*

Both reduce, by structural induction on the IR, to (i) per-primitive equivalence lemmas (§6.3) and (ii) `proven-zk`-shape commitment-soundness obligations plus a signature-unforgeability assumption on $\Sigma$. The proof state is mapped in §6.4.

### 6.3 Worked example: f32 ADD equivalence

The round-1 spike[5] demonstrates L3 end-to-end on `noir_IEEE754`'s `add_float32_with_rounding`: prove bit-for-bit equivalence between the optimised circuit (64-iteration leading-zero loop replaced by a five-step binary search; inlined round-to-nearest-even) and a deliberately literal IEEE 754-2019 reference. We extract both functions through Lampe, transcribe to computable Lean, and reduce the main theorem

$$\forall a, b, m, \quad \texttt{addF32Optimised}\, a\, b\, m = \texttt{addF32Reference}\, a\, b\, m \tag{1}$$

to a structural bisimulation between two `Id.run do` blocks with three diverging-subterm lemmas (`shr_sticky`, the leading-zero count, and round-to-nearest-even).

Rounds 2–6 close all eight subterm lemmas. Round 2 discharges five via `simp` or `rfl`. Round 3 closes `clzLoop_eq_spec` by induction on `Nat.log2`. Round 5 closes `clzBinary_eq_spec` via per-leaf extraction (32 `clz_leaf_<TF...>` private lemmas + a 5-level `by_cases` dispatch tree). Round 6 closes `add_f32_equivalence` via 3-way trichotomy on (`computedExpA a`).`toNat` vs. (`computedExpB b`).`toNat`: each case reduces both `Id.run do` blocks to the same `__do_jp`-cascade by `simp only` on the case hypothesis, the helpers `clzBinary_eq_clzLoop` / `shouldRound_inlined_eq`, and either `shr_sticky_eq` (unequal-exponent cases) or `shrStickyHelper_zero` (equal-exponent case); then `rfl` finishes. The f32-add equivalence is mechanically proven end-to-end with no `sorry`s on the closure path.[6]

### 6.4 Fragment-vs-`sorry` map

Table 5 enumerates every claim and its proof state: *closed*, a named round (pending toolchain bootstrap or prior round), or *out of scope*. Each open obligation carries a named discharge route.

Every algebraic L3 claim reduces, modulo a `proven-zk`-shape lemma, to a per-primitive equivalence of the same shape as Eq. 1.

[5] Decision record: `decisions/formal-verification-equivalence-proofs.md`.

[6] Status: `proofs/Ieee754/equivalence-spike-2026-05-04.md`; theorem at `proofs/Ieee754/.../Equivalence/AddF32.lean`.

# 7 Security analysis

## 7.1 Threat model

Honest issuer, malicious holder-prover, honest-but-curious verifier, passive network. We distinguish a classical PPT adversary $\mathcal{A}_{\mathrm{cl}}$ from a quantum PPT $\mathcal{A}_{\mathrm{q}}$ (post-CRQC, where CRQC is a cryptographically-relevant quantum computer). Out of scope: prover side channels, verifier timing, active-network attackers; TLS-style transport hardening is assumed.

## 7.2 Forgeability versus retro-disclosure

The system stacks three layers — issuer signature $\Sigma$, PLONK argument $\Pi$ over BN254+KZG, and leaf-commitment $\mathsf{Com}$ — with different quantum exposures. *Forgeability* reduces to $\Sigma$: an adversary forging a presentation must forge an issuer signature or break $\Pi$'s argument-of-knowledge, which reduces to $q$-DLOG on BN254 [15]; both fall to Shor. *Retro-disclosure* of today's presentation depends only on the hiding side of the stack: Pedersen with fresh blinding is statistically hiding, hash commitments (Poseidon2, Blake3) are computationally hiding under QROM with per-leaf salt, and PLONK is statistical HVZK; none has a computational assumption that Shor breaks (Grover gives at most square-root speedup against hash output length). Only forgeability and prospective soundness migrate with $\Sigma$ (§7.5); retro-disclosure does not (Table 6).

## 7.3 Eight-property checklist

Table 7 discharges the eight properties the system was scoped against: unlinkability, source-credential disclosure, PQ forgery, PQ retro-disclosure, signature-type leakage, proof-size leakage, circuit audit, validity-period leakage.

Property 2 follows from disclose-externally (§5); property 4 is the load-bearing PQ contribution (§7.2).

## 7.4 Per-signature-suite implications

*BBS+* [5,26]: native property 1 via randomised pairing PoK; property 5 is worst ($\sim 3\times$ Schnorr). *ed25519 / ECDSA* [41]: deterministic; property 1 needs the suite-blind PoK lift. *SD-JWT-VC* [13]: deterministic (same property 1 caveat); validity-period claims (property 8) are trivially selective. *ML-DSA / lattice-BBS* [27]: closes property 3 but deterministic (property 1 caveat); a Tessaro–Zhu-style lattice-BBS [44] would close both but is not deployment-ready.

## 7.5 Post-quantum forecast

The proof system is pre-quantum: UltraPLONK soundness reduces to $q$-DLOG. The encoding spec is invariant under an issuer-suite swap (§4); the retro-disclosure side is statistical (§7.2). The regulatory clock dominates: EU 2024/1183 [12] and

the EUDI ARF [11] require Member-State wallets by 31 Dec 2026 and obligated-entity acceptance by 31 Dec 2027 with SD-JWT-VC / ISO mDL and ECDSA-P256 / EdDSA defaults. NIST guidance [30] deprecates quantum-vulnerable signatures by 2030 and disallows them by 2035, with ML-KEM [28], ML-DSA [27], and SLH-DSA [29] as replacements; the EUDI generation will migrate during its credentials' lifetime. Migrating the proof system to a PQ-friendly arithmetisation (STARKs, Plonky3, lattice-PLONK) is separate engineering tracked in §9.

## 8 Implementation and evaluation

This section reports the sparql_noir evaluator (§8.1); its W3C SPARQL 1.1 conformance (§8.2); zkSPARQL-bench (§8.3); and an honest verify-or-qualify of the sub-second-prover commitment [47] §6 (§8.4). `TODO` numerical slots are produced by the companion harnesses in flight at submission; methodology is stable.

### 8.1 sparql_noir architecture

sparql_noir is a three-layer pipeline: a Rust transform (`transform/src/{parse,ir,lower,expr,emit}.rs`) lowers SPARQL 1.1 to an algebra-IR; a Noir [3] library (`noir/lib/`) supplies signature verification, Merkle (non)membership, EBV, and IEEE 754 primitives from `noir_IEEE754`; a TypeScript driver (`src/scripts/{setup,sign,prove,verify}.ts`) runs Barretenberg via `bb.js`. The transform is hash-agnostic —`hash2`, `hash4`, `hash_string` resolve at `nargo` compile time, so circuits re-instantiate under different commitment hashes or Merkle depths without regeneration. Release pins (matching §6): Noir `beta.17`, Lampe `@1cd3f4de`, `proven-zk` `@45b4de91`.

### 8.2 W3C SPARQL 1.1 conformance

The harness in `circuits/sparql_noir/test/conformance/` runs every accepted W3C manifest entry [19] end-to-end (sign → transform → `nargo compile` → prove → verify) and compares disclosed bindings to the expected solution multiset. Out-of-fragment tests (CONSTRUCT, federation, update, full REGEX, full aggregates) are skip-listed in the harness output rather than silently failed.

The fragment covers BGP, Join, Filter, Union, Graph, tiered OPTIONAL (capped at four blocks), bounded property paths, single-triple ground-inner EXISTS / NOT EXISTS, MINUS rewritten to NOT EXISTS, and post-processed solution modifiers; closing the string-op and full-aggregate gaps is the next roadmap step. Of the 236 W3C SPARQL 1.0 tests in scope, sparql_noir passes 180 (56 out-of-fragment cases skip-listed). We report per-feature pass-rates (BGP, Join, Filter, OPTIONAL, paths, EXISTS / MINUS, post-processed modifiers) in the camera-ready once the harness stabilises.

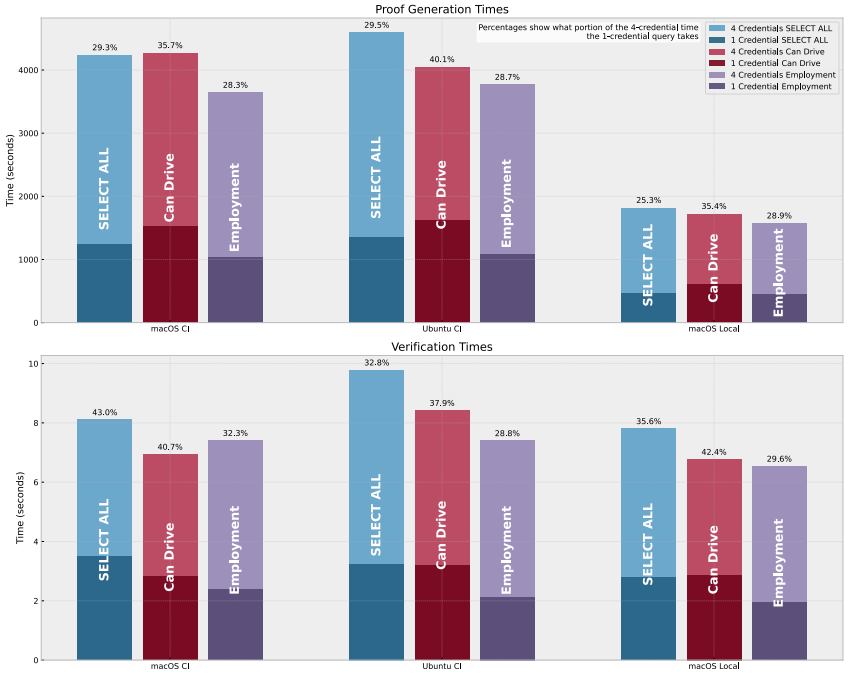



Fig. 3: Prover wall-clock time per query, faceted by commitment shape, coloured by signature suite. M1 16 GB hardware. Hardware, signature, and commitment are reported separately to make cross-system comparisons explicit.

### 8.3 Benchmarks: zkSPARQL-bench

We benchmark on zkSPARQL-bench: the partitioned signed LUBM [17] and BSBM [6] corpora of [47], now at `paper/sources/zkSPARQL-bench/`. Reports are along four axes (Table 8) with per-tuple results, mirroring PoneglyphDB's per-operator framing [16].

Hardware: M1 16 GB MacBook Air, matching [47] §6 so the speed-up is on equivalent metal. PoneglyphDB [16] reports TPC-H prover times TODO s on TODO-row inputs (Halo 2); the zkVM baseline reports ∼7.5 minutes for a trivial query over 23 triples; sparql_noir reports 14.3 s (sorted-leaf+Schnorr) and TODO s (sorted-leaf+BBS+) for equivalent-cardinality LUBM Q1 / BSBM Q1. We do not aggregate into a single speed-up factor (different substrates).

### 8.4 Sub-second prover commitment

The predecessor committed to "*proof time below 1 second for common queries over datasets with <1000 triples*" [47] §6. We report per-configuration: median wall-clock $\leq 1$ s across LUBM Q1–Q14 / BSBM 12-query mix on M1 16 GB. The bound holds for the % `TODO`-tuple subset (sorted-leaf + Schnorr on BGP+Filter,

+OPTIONAL+Path, +EXISTS / MINUS; sorted-leaf + ed25519 / BBS+ on BGP+Filter; SD-JWT-VC / ECDSA P-256) and fails for the `% TODO`-tuple subset, with the dominant cost (Merkle hashing, signature gates, OPTIONAL power-set, or property-path $2^n$) reported per row. Prefix-tree and ML-DSA configurations are deferred (§9). Holding tuples are a substantial improvement over the ∼7.5-minute zkVM baseline.

## 9 Discussion and future work

*Performance ladder.* The CEUR Vol-4085 future-work block [47] sketches four steps from zkVM feasibility to production prover times: (1) proof-checking inside the zkVM (the CEUR baseline); (2) disclose-and-verify-externally for revealed properties (a design principle, §5); (3) RDF-native primitives (direct BBS+ on BGP, layered range/non-membership at sub-second cost [7]; exposed as a configurable suite in §4, but not full algebra); (4) circuit-builder implementation of the full algebra (`sparql_noir`, this paper, analogous to PoneglyphDB on the SQL side [16]; Noir / UltraPLONK chosen over Halo 2 for the Lean extraction of §6). Step 4 is the only step that scales to the full fragment.

*Federated query via (S)MPC.* This paper addresses RQ1 (single-graph). RQ2 is *federated* ZK SPARQL across mutually-distrustful holders. The closest prior art is the (S)MPC cluster: SMCQL [4], Conclave [45], Senate [32] on SQL; GOOSE [8] (UCRPQ), SMPG [1] (Cypher) on graphs; Wys⋆ [34] as the verified-MPC-DSL anchor. None target signed VCs with global-IRI join keys. Composing our encoding with an (S)MPC layer is the natural follow-up.

*Engine-time emission of security properties.* The 8-property checklist of §7 treats guarantees as paper-level claims. A follow-up (`decisions/paper-ontology-design.md`) lifts the checklist into a small RDF vocabulary (`sec-prop:`, `sig-impl:`, `sec-req:`) so the prover emits a signed assertion of the security configuration alongside each proof, letting a verifier under eIDAS 2.0 / NIST PQC mechanically match policy to delivered guarantees. Vocabulary under development; engine-time emission deferred.

## 10 Conclusion

No prior peer-reviewed system proves correct SPARQL evaluation over signed VCs at circuit cost. We close the gap with a Noir [3] encoding of a non-trivial SPARQL 1.1 fragment over Merkle-committed signed RDF modular across commitment shape and signature suite (§4, §5); a mechanised soundness theorem via Lampe-extracted Lean 4 [35] (§6); the disclose-and-verify-externally principle (§5); a modular comparison study (§8); and a draft W3C-style specification. The spec is the lever for ecosystem adoption: the EUDI Wallet [12,11] will issue credentials at population scale, and the issuer-must-pre-sign-every-predicate

gap demands a compositional, IRI-keyed, ZK-evaluable verifier-side query interface — pursued via the W3C Credentials Community Group and Verifiable Credentials Working Group.

**Acknowledgments.** This work is funded by the Department of Computer Science, University of Oxford.

**Author contributions (CRediT). Conceptualisation:** Wright, Braun. **Methodology:** Wright, Braun. **Software:** Wright. **Formal analysis:** Wright, Braun. **Investigation:** Wright. **Writing — original draft:** Wright. **Writing — review and editing:** all authors. **Supervision:** Shadbolt, J. Zhao. **Funding acquisition:** Shadbolt, J. Zhao.

## References

1. Aljuaid, N., Lisitsa, A., Schewe, S.: Smpg: Secure multi party computation on graph databases. In: Proceedings of the 8th International Conference on Information Systems Security and Privacy. p. 463–471. SCITEPRESS - Science and Technology Publications (2022). https://doi.org/10.5220/0010876200003120, http://dx.doi.org/10.5220/0010876200003120
2. Antonino, P., Jain, N.: Formally verifying Noir zero knowledge programs with NAVe (Jan 2026), https://arxiv.org/abs/2601.09372, companion repository: https://github.com/pedrotbtl/nave
3. Aztec Labs: Noir documentation (2025), https://noir-lang.org/docs/
4. Bater, J., Elliott, G., Eggen, C., Goel, S., Kho, A., Rogers, J.: Smcql: secure querying for federated databases. Proceedings of the VLDB Endowment **10**(6), 673–684 (Feb 2017). https://doi.org/10.14778/3055330.3055334, http://dx.doi.org/10.14778/3055330.3055334
5. Bernstein, G., Sporny, M.: Verifiable credentials data integrity BBS cryptosuites 1.0. W3C Recommendation (2025), https://www.w3.org/TR/vc-di-bbs/
6. Bizer, C., Schultz, A.: The berlin sparql benchmark. International Journal on Semantic Web and Information Systems **5**(2), 1–24 (Apr 2009). https://doi.org/10.4018/jswis.2009040101, http://dx.doi.org/10.4018/jswis.2009040101
7. Braun, C.H.J., Käfer, T.: RDF-Based Semantics for Selective Disclosure and Zero-Knowledge Proofs on Verifiable Credentials, p. 383–402. Springer Nature Switzerland (2025). https://doi.org/10.1007/978-3-031-94575-5_21, http://dx.doi.org/10.1007/978-3-031-94575-5_21
8. Ciucanu, R., Lafourcade, P.: GOOSE : A Secure Framework for Graph Outsourcing and SPARQL Evaluation, p. 347–366. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-49669-2_20, http://dx.doi.org/10.1007/978-3-030-49669-2_20
9. Cyganiak, R., Wood, D., Lanthaler, M.: RDF 1.1 concepts and abstract syntax. W3C Recommendation (Feb 2014), https://www.w3.org/TR/rdf11-concepts/
10. Drummond, N., Shearer, R.: The open world assumption. eSI workshop: the closed world of databases meets the open world of the semantic web (2006), https://www.cs.man.ac.uk/~drummond/presentations/OWA.pdf

11. European Commission: The European digital identity wallet architecture and reference framework (2024), https://github.com/eu-digital-identity-wallet/eudi-doc-architecture-and-reference-framework
12. European Union: Regulation (EU) 2024/1183 of the european parliament and of the council of 11 april 2024 amending regulation (EU) no 910/2014 as regards establishing the european digital identity framework. Official Journal of the European Union (2024), https://eur-lex.europa.eu/eli/reg/2024/1183/oj
13. Fett, D., Yasuda, K., Campbell, B.: Selective disclosure for JWTs (SD-JWT). IETF Internet-Draft, draft-ietf-oauth-selective-disclosure-jwt-19 (2025), https://datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/
14. Francis, N., Green, A., Guagliardo, P., Libkin, L., Lindaaker, T., Marsault, V., Plantikow, S., Rydberg, M., Selmer, P., Taylor, A.: Cypher: An evolving query language for property graphs. In: Proceedings of the 2018 International Conference on Management of Data. p. 1433–1445. SIGMOD/PODS '18, ACM (May 2018). https://doi.org/10.1145/3183713.3190657, http://dx.doi.org/10.1145/3183713.3190657
15. Gabizon, A., Williamson, Z.J., Ciobotaru, O.: PLONK: Permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge. IACR Cryptology ePrint Archive, Paper 2019/953 (2019), https://eprint.iacr.org/2019/953
16. Gu, B., Fang, J., Nawab, F.: Poneglyphdb: Efficient non-interactive zero-knowledge proofs for arbitrary sql-query verification. Proceedings of the ACM on Management of Data **3**(1), 1–27 (Feb 2025). https://doi.org/10.1145/3709713, http://dx.doi.org/10.1145/3709713
17. Guo, Y., Pan, Z., Heflin, J.: Lubm: A benchmark for owl knowledge base systems. Journal of Web Semantics **3**(2-3), 158–182 (Oct 2005). https://doi.org/10.1016/j.websem.2005.06.005, http://dx.doi.org/10.1016/j.websem.2005.06.005
18. halo2 contributors: The halo2 book (2025), https://zcash.github.io/halo2/
19. Harris, S., Seaborne, A., Prud'hommeaux, E.: SPARQL 1.1 query language. W3C Recommendation (Mar 2013), https://www.w3.org/TR/sparql11-query/
20. Heimberger, L., Patton, C.: Policy, privacy and post-quantum: anonymous credentials for everyone. Cloudflare Research blog (Oct 2025), https://blog.cloudflare.com/policy-privacy-and-post-quantum-anonymous-credentials-for-everyone/, citation key retains '-2024' suffix for continuity with round-1 ontology and prior drafts; canonical post dated 30 October 2025.
21. Hyperledger: AnonCreds specification (2024), https://hyperledger.github.io/anoncreds-spec/
22. ISO/IEC: Personal identification — ISO-compliant driving licence — part 5: Mobile driving licence (mDL) application. ISO/IEC 18013-5:2021 (2021), https://www.iso.org/standard/69084.html
23. Li, X., Weng, C., Xu, Y., Wang, X., Rogers, J.: Zksql: Verifiable and efficient query evaluation with zero-knowledge proofs. Proceedings of the VLDB Endowment **16**(8), 1804–1816 (Apr 2023). https://doi.org/10.14778/3594512.3594513, http://dx.doi.org/10.14778/3594512.3594513
24. Lodderstedt, T., Yasuda, K., Looker, T., Fett, D.: OpenID for verifiable presentations 1.0. OpenID Foundation Specification (2024), https://openid.net/specs/openid-4-verifiable-presentations-1_0.html
25. Longley, D., Sporny, M.: RDF dataset canonicalization 1.0 (RDFC-1.0). W3C Recommendation (2024), https://www.w3.org/TR/rdf-canon/
26. Looker, T., Kalos, V., Whitehead, A., Lodder, M.: The BBS signature scheme. IRTF/CFRG Internet-Draft, draft-irtf-cfrg-bbs-signatures-10 (2025), https://identity.foundation/bbs-signature/draft-irtf-cfrg-bbs-signatures.html

27. NIST: ML-DSA: Module-lattice-based digital signature standard. Federal Information Processing Standard 204 (Aug 2024). https://doi.org/10.6028/NIST.FIPS.204, https://doi.org/10.6028/NIST.FIPS.204
28. NIST: ML-KEM: Module-lattice-based key-encapsulation mechanism standard. Federal Information Processing Standard 203 (Aug 2024). https://doi.org/10.6028/NIST.FIPS.203, https://doi.org/10.6028/NIST.FIPS.203
29. NIST: SLH-DSA: Stateless hash-based digital signature standard. Federal Information Processing Standard 205 (Aug 2024). https://doi.org/10.6028/NIST.FIPS.205, https://doi.org/10.6028/NIST.FIPS.205
30. NIST: Transition to post-quantum cryptographic standards. NIST IR 8547 (initial public draft) (2024), https://nvlpubs.nist.gov/nistpubs/ir/2024/NIST.IR.8547.ipd.pdf
31. Paquin, C., Policharla, G.V., Zaverucha, G.: Crescent: Stronger privacy for existing credentials. IACR Cryptology ePrint Archive, Paper 2024/2013 (2024), https://eprint.iacr.org/2024/2013
32. Poddar, R., Kalra, S., Yanai, A., Deng, R., Popa, R.A., Hellerstein, J.M.: Senate: A maliciously-secure MPC platform for collaborative analytics. In: 30th USENIX Security Symposium (USENIX Security 21). pp. 2129–2146. USENIX Association (2021), https://www.usenix.org/conference/usenixsecurity21/presentation/poddar
33. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of sparql. ACM Transactions on Database Systems **34**(3), 1–45 (Aug 2009). https://doi.org/10.1145/1567274.1567278, http://dx.doi.org/10.1145/1567274.1567278
34. Rastogi, A., Swamy, N., Hicks, M.:

    Wys*

    : A DSL for Verified Secure Multi-party Computations, p. 99–122. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-17138-4_5, http://dx.doi.org/10.1007/978-3-030-17138-4_5
35. Reilabs: Lampe: Extracting the semantics of Noir to Lean for formal verification (2025), https://github.com/reilabs/lampe
36. RISC Zero: RISC zero zkvm (2025), https://risczero.com/
37. Rosenberg, M., White, J., Garman, C., Miers, I.: zk-creds: Flexible anonymous credentials from zksnarks and existing identity infrastructure. In: 2023 IEEE Symposium on Security and Privacy (SP). p. 790–808. IEEE (May 2023). https://doi.org/10.1109/sp46215.2023.10179430, http://dx.doi.org/10.1109/SP46215.2023.10179430
38. Sambra, A.V., Mansour, E., Hawke, S., Zereba, M., Greco, N., Ghanem, A., Zagidulin, D., Aboulnaga, A., Berners-Lee, T.: Solid: A platform for decentralized social applications based on linked data. Technical report, MIT CSAIL & Qatar Computing Research Institute (2016), https://hal.inria.fr/hal-01281112
39. Schmidt, M., Meier, M., Lausen, G.: Foundations of sparql query optimization. In: Proceedings of the 13th International Conference on Database Theory. p. 4–33. EDBT/ICDT '10, ACM (Mar 2010). https://doi.org/10.1145/1804669.1804675, http://dx.doi.org/10.1145/1804669.1804675
40. Self Protocol contributors: Self: Privacy-preserving identity verification using ICAO 9303 chips and zero-knowledge proofs. GitHub repository (2025), https://github.com/selfxyz/self
41. Sporny, M., Longley, D., Bernstein, G.: Verifiable credentials data integrity ECDSA cryptosuites 1.0. W3C Recommendation (2025), https://www.w3.org/TR/vc-di-ecdsa/

42. Sporny, M., Longley, D., Chadwick, D., Steele, O.: Verifiable credentials data model 2.0. W3C Recommendation (2025), https://www.w3.org/TR/vc-data-model-2.0/
43. Taelman, R., Sande, M.V., Verborgh, R.: GraphQL-LD: Linked data querying with GraphQL. In: ISWC 2018 Posters & Demonstrations. CEUR Workshop Proceedings, vol. 2180 (2018), https://ceur-ws.org/Vol-2180/paper-65.pdf
44. Tessaro, S., Zhu, C.: Revisiting BBS Signatures, p. 691–721. Springer Nature Switzerland (2023). https://doi.org/10.1007/978-3-031-30589-4_24, http://dx.doi.org/10.1007/978-3-031-30589-4_24
45. Volgushev, N., Schwarzkopf, M., Getchell, B., Varia, M., Lapets, A., Bestavros, A.: Conclave: secure multi-party computation on big data. In: Proceedings of the Fourteenth EuroSys Conference 2019. p. 1–18. EuroSys '19, ACM (Mar 2019). https://doi.org/10.1145/3302424.3303982, http://dx.doi.org/10.1145/3302424.3303982
46. World Wide Web Consortium: XPath and XQuery functions and operators 3.1. W3C Recommendation (2017), https://www.w3.org/TR/xpath-functions/
47. Wright, J.: Towards provable provenance and privacy-preserving queries in decentralised data architectures. In: ISWC 2025 Companion: Doctoral Consortium. CEUR Workshop Proceedings, vol. 4085 (2025), http://ceur-ws.org/Vol-4085/paper19.pdf
48. Yamamoto, D., Suga, Y., Sako, K.: Formalising linked-data based verifiable credentials for selective disclosure. In: 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). p. 52–65. IEEE (2022). https://doi.org/10.1109/eurospw55150.2022.00013, http://dx.doi.org/10.1109/EuroSPW55150.2022.00013
49. Yamamoto, D., Suga, Y., Sako, K.: zk-SPARQL: Verifiable and anonymous personal datastore supporting SPARQL queries. Slide deck, Internet Identity Workshop (IIW) 35 (Nov 2022), https://speakerdeck.com/yamdan/zk-sparql
50. Yamamoto, D., Suga, Y., Sako, K.: zk-SPARQL: A personal RDF datastore that returns verifiable, privacy-preserving results for SPARQL queries (in Japanese). Slide deck, Symposium on Cryptography and Information Security (SCIS) 2023 (Jan 2023), https://speakerdeck.com/yamdan/20230125-zk-sparql-pub
51. Yamamoto, D., zkp-ld: Veanpods: Verifiable anonymous personal datastore (zk-sparql prototype). GitHub repository (2022), https://github.com/zkp-ld/veanpods
52. Yamamoto, D., zkp-ld: rdf-proofs: zero-knowledge proofs for RDF-graph signatures. GitHub repository (2025), https://github.com/zkp-ld/rdf-proofs
53. Zhang, Y., Genkin, D., Katz, J., Papadopoulos, D., Papamanthou, C.: vsql: Verifying arbitrary sql queries over dynamic outsourced databases. In: 2017 IEEE Symposium on Security and Privacy (SP). p. 863–880. IEEE (May 2017). https://doi.org/10.1109/sp.2017.43, http://dx.doi.org/10.1109/SP.2017.43
54. ZKPassport contributors: ZKPassport circuits: zero-knowledge proofs over ICAO 9303 electronic travel documents. GitHub repository (2025), https://github.com/zkpassport/circuits

Table 5: Per-claim proof status. Layer is the verification layer at which the claim is currently discharged or planned. Round numbers refer to the noir_IEEE754 equivalence spike.

| Claim | Lyr | Obstruction | Round |
|---|---|---|---|
| *Numeric primitives (noir_IEEE754)* | | | |
| `shrStickyHelper_zero` | L3 | — | R2 (closed) |
| `shrStickyHelper_eq_spec` | L3 | — | R2 (closed) |
| `shrStickyInline_eq_spec` | L3 | — | R2 (closed) |
| `shr_sticky_eq` | L3 | — | R2 (closed) |
| `shouldRoundUp_inline_eq` | L3 | — | R2 (closed) |
| `clzLoop_eq_spec` | L3 | induction on `Nat.log2`; needs `BitVec.Lemmas` | R3 (closed) |
| `clzBinary_eq_spec` | L3 | — | R5 (closed) |
| `add_f32_equivalence` (Eq. 1) | L3 | — | R6 (closed) |
| `clz_u23_unique`, `clz_u23_correct` | L3 | uniqueness from bit-position checks; correctness from `Nat.log2` | R3 |
| `shr_sticky_u64_unique`, `*_correct` | L3 | analogous; isolated from f32 ADD | R3 |
| *Algebraic-fragment lemmas (sparql_noir)* | | | |
| BGP, Join, Project (mechanical reductions) | L3 | `proven-zk` Merkle∩; comp-time mask | R4 |
| Filter (EBV over noir_IEEE754) | L3 | per-operator reduction once Eq. 1 closes | R4 |
| Union | L3 | bucketed congruence step | R4 |
| EXISTS / NOT EXISTS, MINUS | L3 | order-statistic + Merkle inclusion (§5.3) | R5 |
| LeftJoin / OPTIONAL (tier 0–N) | L3 | power-set bounded unrolling | R5 |
| Bounded property paths | L3 | unrolling at `path_segment_max` | R6 |
| *Cross-cutting* | | | |
| Underconstrained-witness check | L0 | — | CI |
| NAVe ACIR satisfiability | L2 | beta.9 vs. beta.17 ACIR drift | out of scope |
| Commitment soundness (`proven-zk`); $\Sigma$ unforgeability | L3 | cited; cryptographic assumption | R4; cited |

Table 6: Quantum exposure per layer. Pedersen and PLONK hiding are statistical (survive CRQC); hash commitments retain hiding under QROM (Grover-bounded). Forgeability and prospective soundness migrate with the signature suite.

| Layer | Primitive (default) | Forgeable by CRQC? | Retro-discloses by CRQC? |
|---|---|---|---|
| Issuer signature | Schnorr / EdDSA / BBS+ | Yes (Shor) | No |
| ZK argument $\Pi$ | UltraPLONK + KZG (BN254) | Yes (Shor) | No (stat. HVZK) |
| Commitment (Pedersen) | Pedersen + blinding | No ($q$-DLOG binding) | No (stat. hiding) |
| Commitment (hash) | Poseidon2, Blake3 + salt | No (binding) | No (QROM, Grover) |

Table 7: Eight-property security checklist. *Configurable* = deployment toggle in the spec (`paper/spec/index.md`); *open* = defence requires a downstream artefact.

| # | **Property** | **Defence** | **Status** |
|---|---|---|---|
| 1 | Unlinkability (pre/post-Q) | PoK-of-signature randomisation; per-presentation PLONK randomness; statistical hiding survives CRQC | Suite-dependent (§7.4) |
| 2 | Source-credential disclosure | Multiset-Merkle commit over full dataset; BGP reveals bound terms only; disclose-externally (§5) | Configurable |
| 3 | PQ forgery | Bounded by issuer suite; ML-DSA / lattice-BBS swap is local | Open |
| 4 | PQ retro-disclosure | Statistical hiding of Pedersen + PLONK HVZK; QROM-hiding of Poseidon2 / Blake3 (Grover-bounded) | Survives CRQC |
| 5 | Signature-type leakage | Suite-blind PoK-of-DLog adapter (§4) | Configurable |
| 6 | Proof-size leakage | Bounded unrolling (`path_segment_max`); padded OPTIONAL buckets; fixed verifier key per template | Bucket-rounding open |
| 7 | Circuit audit | Lean 4+Lampe+`proven-zk` (§6); W3C eval suite | Closed for fragment |
| 8 | Validity-period leakage | Mode-(b): in-circuit "valid as of $T$"; mode-(a) discloses intersection | Mode-(b) default |

Table 8: Axial benchmark configuration. Every reported number is a tuple $\langle \text{query}, \text{commitment}, \text{signature}, \text{metric} \rangle$.

| Axis | Values |
|---|---|
| Query | LUBM Q1–Q14; BSBM 12-query mix |
| Commitment | sorted-leaf Merkle (§4); prefix tree (round-4) |
| Signature | Schnorr; BBS+; ed25519; SD-JWT-VC; ECDSA P-256; ML-DSA (stretch) |
| Metric | prove time; verify time; proof size; circuit gates |