Sombra: A Quantum-Safe Private Payments Protocol on Solana

Bonsol Labs

Bonsol Labs

Abstract

We present *Sombra*, a private payments protocol on Solana built post-quantum at genesis. In prior work [Bon26b; Bon26a], we argued that post-quantum migration for privacy chains faces structural constraints—historical ciphertexts cannot be recalled, address linkability becomes permanent, migration windows create coordination problems—that may be unsolvable for protocols not built quantum-safe from the start. Sombra is the constructive answer.

The protocol uses STARK proofs as its proof system, Kyber-768 (NIST ML-KEM) for note encryption, and a post-quantum native key hierarchy; spend authorization is constructed as a STARK-based zero-knowledge proof of decryption rather than a digital signature. These choices eliminate the quantum attack vectors compromising classical shielded designs [BZG+26] by primitive choice rather than future migration. Sombra deploys as a standard Solana program with no consensus changes.

We specify the construction, name the on-ramp boundary and wrapped-token provenance as known limitations, position the work against prior privacy designs on Solana and elsewhere, and outline a vault-less quantum-native token as the path to closing the remaining perimeter gap.

1 Introduction

In two recent analyses, we argued that post-quantum migration for privacy chains faces structural constraints that may be unsolvable for protocols not built quantum-safe from the start [Bon26b; Bon26a]. Historical ciphertexts cannot be recalled. Diversified-address linkability is permanent. Migration windows create race conditions between legitimate users and quantum-equipped attackers, with detection of the moment a cryptographically relevant quantum computer (CRQC) arrives constituting its own coordination problem. Each of these damages is backward-looking: the encrypted data, the published addresses, and the committed proofs are already on a public, immutable ledger by the time a CRQC is announced. We closed those analyses with an open question—does post-quantum private payments require quantum safety from the start?—and present *Sombra*, a private payments protocol on Solana built post-quantum at genesis, as the constructive answer.

The urgency is no longer hypothetical. Across 2025–26, on-chain privacy emerged as a defining priority: the Tornado Cash delisting, the Ethereum Foundation's dedicated Privacy Cluster, the enshrinement of privacy as a core principle of the Ethereum protocol, and the outperformance of privacy assets all confirmed a market-wide repricing of confidentiality. Two analyses crystallized the threat. Justin Thaler [Tha25] identified privacy chains as the *exception* among blockchains—the one category for which harvest-now, decrypt-later (HNDL) attacks are not hypothetical but an active campaign against present-day users. Google Quantum AI [BZG+26] then provided independent technical validation, lowering

Email: contact@bonsol.org (Bonsol Labs)

Date of this document: 2026-04-20.

the quantum-resource estimate for breaking elliptic-curve cryptography by an order of magnitude and supplying a taxonomy of distinct attack vectors against EC-based shielded protocols.

The threat described above generalizes to every privacy chain, but it has not yet produced a response on Solana. Solana's existing privacy designs—Umbra (Arcium), built on multi-party computation [Arc]; the Token-2022 Confidential Transfers extension, built on ElGamal encryption with classical zero-knowledge proofs [Sol]; the now-deprecated Elusiv; and Otter Cash, also a classical-ZK shielded transfer protocol—are each substantive engineering efforts in their own terms. None has a published post-quantum posture. Each places encrypted state on a public, immutable ledger using elliptic-curve-derived primitives, exposing the same harvest-now, decrypt-later surface that the broader privacy-chain category faces. Solana's high throughput and low fees make it an attractive substrate for private payments at scale; the absence of a quantum-safe option is the gap this paper addresses.

Sombra is a private payments protocol on Solana built post-quantum at genesis. The protocol uses STARKs as the underlying proof system, Kyber-768 (NIST ML-KEM) for note encryption, and a post-quantum native key hierarchy; spend authorization is constructed as a STARK-based zero-knowledge proof of decryption rather than a digital signature. These choices eliminate the quantum attack vectors that compromise classical shielded-protocol designs [BZG+26], by primitive choice rather than by future migration. Sombra deploys as a standard Solana program with no consensus changes. We acknowledge two known limitations of the current design: deposits from external Solana wallets reveal an external-wallet-to-vault link, and assets bridged from chains with classical signature schemes inherit their source-chain quantum exposure. The remainder of the paper specifies the threat model (§2), the construction (§3), and the positioning against prior work (§4); states the tradeoffs we accept (§5); and outlines the path to closing the remaining perimeter gap (§6).

2 Threat Model

2.1 Adversary

We model a *future quantum-equipped network adversary*. The adversary is **network-observing**: they see all on-chain data, including every encrypted note, every transaction, and every commitment posted to Solana since Sombra's first block. They are **active**: they may issue valid Solana transactions and interact with Sombra normally as a participant. They are **archival**: they collect on-chain data today and store it indefinitely, instantiating the harvest-now, decrypt-later premise that motivates the design [BZG+26]. And they are **quantum-equipped at some future time** T: at T they possess a cryptographically relevant quantum computer capable of running Shor's algorithm at scale on elliptic-curve-sized inputs.

The adversary is bounded by the hardness assumptions Sombra builds on. Specifically, we assume they cannot violate (i) the NIST-standardized lattice hardness assumptions underlying ML-KEM/Kyber-768—concretely, Module-LWE [NIS24b]; (ii) the collision-resistance of standardized hash functions used in the construction (SHA-3 and Poseidon); and (iii) the soundness of properly-instantiated STARK protocols, which reduces to (ii) and the security of FRI in the random-oracle model. Every security claim in the rest of this paper is conditional on these three assumptions and on Solana's L1 consensus.

2.2 Security Goals

Against the adversary defined above, Sombra targets the following five properties:

1. **Confidentiality.** Transfer amounts, recipients, and memos remain hidden, including against retroactive decryption attempts at time T applied to ciphertexts archived before T.

2. **Unlinkability.** Distinct vaults controlled by the same user remain unlinkable from on-chain data alone.

3. **Soundness.** No value can be minted into a vault that was not legitimately deposited or received.

4. **Spend authorization.** Only the holder of the relevant post-quantum key material can spend a note.

5. **Transfer-graph privacy.** The graph of who-paid-whom cannot be reconstructed from on-chain data.

2.3 Out of Scope

The following exposures are not addressed by the cryptographic core of the protocol:

1. **On-ramp identity link.** MVP1 deposits from an external Solana wallet into a Sombra vault reveal the external-wallet-to-vault correspondence on-chain. This is a known perimeter gap; §3.5 discusses it directly and §6.1 describes the mitigation we are pursuing.

2. **Wrapped-token provenance.** Assets bridged into Sombra from chains with classical signature schemes carry their source-chain quantum exposure; a CRQC that forges provenance on the source chain forges it into Sombra. The mitigation is a quantum-native asset issued inside Sombra (§3.5, §6.1).

3. **User-side key compromise.** Leaked seed phrases, malicious wallet software, and stolen keys are out of scope for any cryptographic protocol; Sombra inherits the standard endpoint-security caveats.

4. **Side-channel and physical attacks.** Power analysis, timing leaks, fault injection, and physical compromise of user devices or proving infrastructure are addressed at the implementation layer, not by the protocol design.

5. **Solana consensus failure.** Sombra deploys as a standard Solana program and inherits Solana's L1 security model; an L1 compromise compromises Sombra.

3 The Sombra Protocol

3.1 Vault Model and Account Abstraction

A Sombra *vault* is a per-user shielded account managed by the Sombra Solana program. It holds the user's set of post-quantum-encrypted UTXOs and the public key material used to authorize spends from that set. All state internal to a vault—note ciphertexts, commitments, and nullifier history—lives on Solana but is opaque to anyone other than the vault owner; the program enforces the protocol's accounting invariants without learning the contents of any note.

The vault is distinct from the user's external Solana wallet. The external wallet—an ordinary Solana account, signed with the chain's classical Ed25519 signature—is used only for the on-ramp deposit that funds the vault and for the destination signature on the off-ramp. Once funds are inside, the vault is operated entirely through post-quantum key

material; the external wallet has no further authority over vault state and does not appear in any internal transfer.

Vault creation is paired with the user's first contract interaction in a single Solana transaction that both initializes the vault account and posts the initial commitment to the user's PQ public key. We adopt this single-transaction pattern as a recurring architectural decision throughout the protocol—it appears again in the transfer construction (§3.3)—because it removes the class of failure modes in which a partially-initialized object is observable on-chain between two transactions.

The vault's PQ key material is derived from the user's seed phrase via a standard derivation-path scheme with an incrementing counter, in the spirit of BIP-32. This replaces an earlier MVP design in which on-ramp keys were sampled at random and held only in transient client state—an approach that admitted a fund-loss failure mode if the client lost the key before the deposit was spent. Recovery logic that traverses the derivation tree to locate vaults associated with a given seed is deferred to a post-V1 release; the V1 protocol fixes the derivation so that recovery is well-defined when that work lands.

The on-chain vault state is a set of Kyber-768 ciphertexts, one per unspent note, together with the commitments and nullifier history needed to enforce no-double-spend. Ownership of any individual note is established not by a signature against the vault's public key but by the zero-knowledge proof of decryption construction (§3.2.3); the vault account itself stores no spending authority that an attacker could exfiltrate by compromising on-chain state alone.

3.2 Cryptographic Primitives

Sombra's cryptographic design rests on three post-quantum primitives—STARK proofs, Kyber-768 encryption, and a post-quantum native key hierarchy—and one distinctive construction, the zero-knowledge proof of decryption, which replaces digital signatures for spend authorization. We describe each in turn, stating what it does, why it is the right choice for Sombra, and which attack vector from [BZG+26] it eliminates.

3.2.1 STARK proofs as the proof system

Sombra uses STARKs (Scalable Transparent ARguments of Knowledge) as its underlying proof system. STARK soundness reduces to the collision-resistance of standardized hash functions and the security of FRI; it requires no trusted setup and uses no elliptic-curve assumptions.

In Sombra, STARKs serve two roles: the transfer circuit (§3.3) proves that an output set of notes is a valid transformation of an input set, and the spend-authorization construction (§3.2.3) proves that the prover can decrypt a target ciphertext. Both roles are downstream of hash-function security alone.

Closes attack vector: proof-system soundness collapse. A cryptographically relevant quantum computer breaks the elliptic-curve discrete-logarithm problem underlying the polynomial commitments in IPA-based SNARKs [BZG+26], enabling forged proofs of arbitrary statements—including proofs that mint counterfeit value. STARK soundness, resting on hash-function security and the random-oracle model, is unaffected.

3.2.2 Kyber-768 for note encryption

Sombra encrypts notes under Kyber-768, the NIST-standardized post-quantum key encapsulation mechanism (ML-KEM, FIPS 203 [NIS24b]). Kyber-768 targets NIST security category 3, with security resting on Module-LWE hardness. Public keys are 1184 bytes and ciphertexts 1088 bytes; §5 discusses the size implications.

Each note in a Sombra transfer is encrypted under the recipient's Kyber-768 public key. The encryption layer carries the note plaintext—amount, asset type, and metadata used by the spend-authorization construction in §3.2.3—and no key material derived from elliptic-curve operations participates in the encapsulation or the symmetric envelope.

Closes attack vector: retroactive decryption of on-chain ciphertexts via the ECDH key exchange. Conventional shielded protocols derive symmetric encryption keys from an elliptic-curve Diffie-Hellman exchange between sender and recipient, with the ephemeral public key recorded on-chain. A future CRQC recovers the encryption key from the on-chain ephemeral and decrypts every historical note payload [BZG+26]. Sombra's encryption has no intermediate elliptic-curve exchange.

3.2.3 Zero-knowledge proofs of decryption

Sombra's distinctive move is to replace the digital signature traditionally used for spend authorization with a zero-knowledge proof of decryption. To spend a note, the prover constructs a STARK proof that they possess a Kyber-768 secret key that decrypts the target ciphertext to a valid note plaintext, without revealing either the secret key or the plaintext.

An alternative path—signing transfers with a NIST-standardized post-quantum signature such as ML-DSA (FIPS 204 [NIS24a])—would also be quantum-safe. We do not take it: a signature design retains a separate primitive with its own forgery surface and requires a second signature (the *binding signature*) to enforce balance conservation. The proof-of-decryption construction collapses both roles into the same STARK proof.

Closes attack vectors: spend-authorization forgery and binding-signature forgery, both of which depend on the unforgeability of an elliptic-curve signature scheme that Shor's algorithm breaks [BZG+26]. Sombra has no signature scheme to forge; spend authorization and balance enforcement both reduce to the soundness of the underlying STARK proof system (§3.2.1).

3.2.4 Post-quantum native key hierarchy

Sombra's key hierarchy contains no elliptic-curve operations. The user's master secret is a high-entropy seed; from it, the protocol derives a Kyber-768 keypair for note encryption and a separate spending key for authorizing ZK proofs of decryption. All derivations use hash-based PRFs or lattice-friendly constructions.

Diversified addresses—multiple addresses derived from the same underlying account, used to compartmentalize financial activity—are constructed in Sombra by hash-based diversification of the recipient's public key material. The diversifier is a per-address public input; it carries no elliptic-curve point and creates no discrete-logarithm relationship between two diversified addresses of the same user.

Closes attack vector: diversified-address linkability. Conventional shielded designs derive diversified addresses through elliptic-curve operations, allowing a CRQC to invert the derivation and link all of a user's addresses retroactively, with no on-chain interaction and no detection [BZG+26]. Sombra has no analogous derivation to invert; the vector is closed by the structure of the key hierarchy itself.

3.3 Transfer Construction

A Sombra transfer follows a UTXO model. The sender consumes one or more input notes from their vault and produces one or more output notes addressed to recipient vaults; balance conservation—the requirement that input value equal output value, modulo a transparent fee—is enforced inside the STARK transfer circuit (§3.2.1) rather than by any on-chain check against plaintext amounts. Each consumed input contributes a nullifier

derived from the note's secret material; the program rejects any transfer whose nullifiers collide with the on-chain history, closing the double-spend surface without revealing which prior note is being spent.

A transfer is submitted as a single Solana transaction that consolidates batch attestation, the combined ciphertext-and-commitment actions for every output note, and the aggregated quorum vote from the attesting nodes. We adopt this atomic single-transaction pattern—first introduced for vault initialization in §3.1—for three reasons: it gives us consistency by construction, since there is no intermediate state in which an output ciphertext is on-chain without its commitment or vice versa; it simplifies the node-side quorum-tracking logic, since attestation and finalization share the same transaction boundary; and it is materially cheaper on Solana's fee schedule than the equivalent multi-transaction sequence.

The same transfer circuit serves both internal transfers and off-ramps. We do not differentiate the two paths inside the prover; the disambiguation is carried by an on-chain contract event. An off-ramp begins by producing a UTXO whose output binding marks it as a withdrawal target, and an attestation-quorum event emitted at finalization carries the context that distinguishes a settled internal transfer from a settled off-ramp. The attesting node never tracks transfer-flow state across transactions: at any moment its responsibility is bounded by the contents of the current transaction and the on-chain event log.

Finalization occurs when the attestation quorum has been reached and the corresponding finalize transaction commits the aggregated vote on-chain. At that point the input notes are nullified, the output notes are durably recorded as part of their recipients' vault state, and any off-ramp branch releases the underlying asset to the named destination. The protocol provides no notion of partial finalization—a transfer is either reversed before quorum or atomically settled.

3.4 Privacy Strategy

Every Sombra transfer pads its consumed-input set with seven to eight *decoy UTXOs* drawn from the on-chain note set. The decoys are indistinguishable on-chain from real inputs: each contributes a commitment opening proven inside the same STARK circuit, and an external observer cannot tell which member of the input set carries the spent value. A fixed decoy count gives every transfer the same on-chain shape and contributes a uniform per-transfer anonymity-set baseline that compounds across the user's transfer history.

We deliberately do not expose a user-facing privacy slider over the decoy count. Two reasons. First, presenting the choice in the UI invites users to reduce it—usually for a fee or latency saving they do not understand the cost of—and the resulting weak-anonymity-set transfers degrade the privacy of every user, not only the user who chose the lower setting. Second, surfacing privacy mechanics in the user interface itself communicates, to anyone observing the user, that an explicit privacy decision is being made. The decoy count is a protocol parameter, not a user preference.

Self-transfer compaction is the one user-facing privacy operation we do expose. The proving time for a Sombra transfer scales with the number of unspent notes the user holds, because each must be considered as a candidate input. A user who has received many small notes can collapse them into a single return note by sending a transfer to their own vault; the operation incurs a normal transfer cost but bounds the worst-case proof time of all subsequent transfers. We treat compaction as a routine maintenance action rather than as a hidden optimization.

3.5 The On-Ramp / Off-Ramp Boundary

A V1 deposit into a Sombra vault is signed by the user's external Solana wallet, and the deposit transaction records that wallet's public key on-chain alongside the destination vault. The link between the external identity and the vault is therefore observable and

permanent: anyone with access to the chain history can correlate the funding wallet with the vault that it created or topped up. Sombra's privacy guarantees apply to every transfer that follows the deposit; they do not apply to the deposit itself. We state this as a scope boundary rather than as a property to be patched: the cryptographic core of the protocol begins at the vault.

A second boundary issue applies to wrapped tokens. Assets bridged into Sombra from a chain whose signature scheme is classical—which today includes essentially every Layer 1 of consequence—carry their source-chain quantum exposure into the vault. A future quantum-equipped attacker can forge a signature on the source chain, construct a provenance record consistent with the bridge's verification logic, and submit a "valid" ownership proof into a Sombra vault. The Sombra circuits would accept it, because Sombra's primitives have no way to distinguish a legitimate cross-chain transfer from a quantum-forged one once the source-chain signature has been compromised. The exposure is a property of the wrapped-asset model, not of the Sombra construction.

Off-ramps reveal the destination wallet by symmetry with the on-ramp: the withdrawal transaction names the recipient's external Solana account, and an observer correlating that account against the chain history can re-identify the off-ramping vault. The mitigation we are pursuing for both the wrapped-token and the off-ramp boundary is the introduction of a quantum-native asset issued directly inside Sombra, minted under a zero-knowledge proof of post-quantum key ownership and not backed by any external asset. Such an asset would have no provenance to forge on a source chain and no canonical destination outside Sombra; transfers in it would not require crossing the boundary at all. We sketch the construction in §6.1.

4 Related Work

4.1 Privacy Chains and the PQ Migration Challenge

Zcash and Project Tachyon represent the most sophisticated post-quantum migration effort underway in a production privacy chain. The published roadmap—Quantum Recoverability, a PIR layer combined with ML-KEM, and the Tachyon and Ragu constructions—is credible engineering [Bow25; Pro; Bz26]. We treat the long-form analysis of what that roadmap can and cannot achieve in two prior Bonsol Labs articles [Bon26b; Bon26a] and do not re-prosecute it here. The summary conclusion of that work is twofold: the roadmap addresses transactions issued after migration but cannot recall historical ciphertexts already on the immutable ledger, and the diversified-address linkability vector is not closed by any element of the published plan.

Aleo, Penumbra, and Aztec are the other major shielded-protocol designs in production or near-production today. None has published a post-quantum migration plan with specified cryptographic primitives or a deployment timeline. Each inherits the same structural problem the Zcash analysis surfaces: classical-encrypted state on an immutable public ledger accrues harvest-now, decrypt-later liability for as long as that state exists.

The pattern across this category is uniform: post-quantum migration is treated as a future retrofit, with classical-encrypted history accumulating in the meantime. Sombra's choice to ship post-quantum-native at genesis is a different point in the design space—not a rebuttal to any specific protocol's roadmap, but the constructive answer to the architectural question raised in our prior work.

4.2 Solana-Native Privacy Approaches

Solana hosts several privacy designs, summarized along two axes—underlying privacy approach and post-quantum posture—in Table 1. We briefly characterize each below.

Table 1: Privacy protocol landscape across two axes: PQ posture and on-chain ciphertext lifecycle. Sombra is the only protocol PQ-native at genesis on a high-throughput chain.

Protocol	**Chain**	**Privacy approach**	**PQ posture**	**On-chain ciphertext lifecycle**
Zcash (Orchard)	Zcash L1	Shielded UTXO + Halo 2 SNARKs	Migration via Tachyon (no date)	Permanent classical-encrypted history
Aleo	Aleo L1	ZK execution model	Migration plan unspecified	Permanent classical-encrypted history
Penumbra	Cosmos	Shielded everything	Migration plan unspecified	Permanent classical-encrypted history
Aztec	Ethereum L2	ZK rollup with privacy	Migration plan unspecified	Permanent classical-encrypted history
Umbra (Arcium)	Solana	MPC-based confidential computation	None stated	N/A (off-chain computation)
SPL-2022 Conf. Transfers	Solana	ElGamal + ZK	None stated	Permanent classical-encrypted history
Sombra	**Solana**	**Shielded UTXO + STARKs + Kyber-768 + ZK-PoD**	**PQ-native at genesis**	**No classical ciphertext liability**

None has a published post-quantum posture.

Umbra, by Arcium, is a substantive engineering effort that uses multi-party computation to perform confidential computation off-chain [Arc]; it is the most architecturally distinct of the four, in that it does not produce on-chain encrypted state of the kind that the harvest-now, decrypt-later threat targets. The Token-2022 Confidential Transfers extension, maintained by the Solana Foundation, provides confidential balance transfers using ElGamal encryption together with classical zero-knowledge proofs [Sol]. Elusiv was a zero-knowledge shielded transfer protocol, now deprecated. Otter Cash is a classical-zero-knowledge shielded transfer protocol in the same general family.

Sombra is the only protocol in this set with a stated post-quantum posture. The two-axis positioning that Table 1 makes structurally legible is the operative point: among Solana privacy protocols, Sombra is the only post-quantum-native one; among post-quantum-aware privacy designs anywhere, Sombra is the only one post-quantum-native at genesis. The moat is geometric, not adversarial—a function of which corner of the design space is occupied—and we present it as such.

4.3 Cryptographic Prior Art

Sombra introduces no new cryptographic primitives. It composes well-studied post-quantum-safe building blocks: STARK proofs, in the line that begins with Ben-Sasson, Bentov, Horesh, and Riabzev's foundational construction [BBHR18]; Kyber-768, the NIST-

standardized module-lattice key-encapsulation mechanism (ML-KEM, FIPS 203 [NIS24b]); and a key hierarchy built from hash-based and lattice-friendly derivations (§3.2.4). The broader research programs on lattice-based zero-knowledge, transparent proof systems, and recursive STARK constructions form the context the design draws on, and we make no claim of novelty at the primitive layer.

The contribution is the composition. The choice to compose these three primitives into a shielded-UTXO protocol whose spend authorization is a zero-knowledge proof of decryption (§3.2.3), deployed as a Solana program with no consensus changes, is what is new here. The primitives are the field's; the construction is ours.

5 Limitations and Tradeoffs

Sombra's PQ-native-at-genesis design accepts three concrete tradeoffs that classical-cryptography designs do not. We name them explicitly here—larger proofs, larger ciphertexts, and a maturity gap relative to long-running elliptic-curve schemes. The honesty signal matters more than diplomatic hedging; we treat each cost as a real cost we pay to avoid a different and worse one.

5.0.1 STARK proof sizes

STARK proofs are larger than equivalent SNARK proofs. A typical SNARK transfer proof is on the order of a few hundred bytes; a STARK proof for the same statement is on the order of tens to hundreds of kilobytes. We accept the size cost in exchange for transparent setup, post-quantum soundness, and elimination of trusted-setup ceremony risk (§3.2.1).

Concrete numbers for Sombra's specific transfer proof are not yet stable enough to publish here. We will report concrete proof-size measurements for Sombra's instantiation in the released implementation documentation; here we report only the order of magnitude.

5.0.2 Kyber-768 ciphertext and key sizes

Kyber-768 has 1184-byte public keys and 1088-byte ciphertexts [NIS24b]. Compared to the elliptic-curve equivalents used in conventional shielded designs—approximately 32-byte public keys and 64-byte ciphertexts—this is roughly a 30× growth in the encryption-layer footprint. Each note ciphertext written on-chain in Sombra is approximately 1 KB rather than approximately 64 bytes.

We accept this footprint as a one-time cost of post-quantum encryption. The off-chain ledger storage direction sketched in §6 stores only a 32-byte commitment to each ciphertext on-chain, with the payload itself stored off-chain—eliminating the on-chain footprint at the cost of an additional storage layer. We do not commit to that mitigation in V1.

5.0.3 Maturity gap

Kyber-768 was standardized as ML-KEM (FIPS 203) in 2024; it is newer than the elliptic-curve primitives that have been continuously analyzed since the 1980s. STARK constructions, recursive proof systems, and lattice-friendly key derivation likewise have less production runtime than their classical counterparts.

We treat this as a real cost. Newer primitives have less battle-testing, and the discovery of a structural weakness in ML-KEM or in a deployed STARK construction would affect Sombra more directly than it would affect a protocol with a smaller post-quantum surface area. We accept the risk because the alternative—building on classical primitives that are known to be quantum-broken [BZG+26]—accumulates a different and worse class of permanent damage.

Across all three tradeoffs, the framing is the same: we would rather solve the performance and maturity problem than the migration problem. The performance and maturity problems admit incremental engineering solutions; the migration problem, once classical-encrypted state is on an immutable ledger, does not.

6 Future Work

6.1 Vault-less Quantum-Native Token Mint

Section 3.5 acknowledged that assets bridged into Sombra from chains with classical signature schemes inherit their source-chain quantum exposure: a quantum-equipped attacker could forge provenance on the source chain and submit a "valid" ownership proof into a Sombra vault. The mitigation we are pursuing is to remove the dependency on external assets entirely for privacy-critical use cases, by introducing a quantum-native token minted directly inside Sombra under proof of post-quantum key ownership, with no external-asset backing.

The construction we are formalizing requires mint operations to be authorized by a zero-knowledge proof analogous to the spend-authorization construction in §3.2.3, extending the post-quantum perimeter to the asset-issuance layer. Several design questions remain open—mint authorization in the transfer circuit, governance over issuance, interaction with the unmodified deposit path—and the work is in active formalization. We do not commit to a deployment timeline in this paper.

6.2 Distributed Prover Network

Sombra's transfer proofs are computed by the user's wallet in the canonical case and, optionally, by remote provers for users on constrained devices. With current GPU-based proving achieving second-scale latencies, the dominant component of end-to-end transfer latency for users far from a prover is the network round-trip—approximately 200–250 ms cross-continent under optimal routing. Once compute-side latency falls below this floor, further improvements to user-perceived performance require placing prover capacity geographically close to users.

We expect Sombra to evolve toward a distributed prover network of geographically-distributed points of presence, with users routed to the nearest prover by latency. The protocol-level requirements are minimal—provers are stateless with respect to the protocol, and there are no consensus changes—so the deployment can grow incrementally without coordination beyond the participating prover operators.

6.3 Off-Chain Ledger Storage

The current Sombra design records full Kyber-768 ciphertexts on-chain, which is the dominant on-chain storage cost. A v2 direction stores only a 32-byte commitment to each ciphertext on-chain, with the ciphertext payload itself stored off-chain—in a decentralized storage layer or a dedicated indexing service. Verification logic remains identical (the on-chain commitment binds the ciphertext); the change is purely a storage-layer optimization.

Beyond the cost reduction, off-chain ciphertext storage opens additional privacy properties: an off-chain layer can be configured to serve note ciphertexts only to the holder of the corresponding viewing key, raising the bar for chain-wide ciphertext harvesting that the current on-chain approach exposes. This direction is deferred to a future protocol revision; we note it here because it materially affects the long-run economic and privacy profile of the system.

6.4 Beyond Payments

The architectural question raised in our prior work [Bon26b; Bon26a] was framed in terms of private payments, but it generalizes. Any privacy primitive that places encrypted state on a public, immutable ledger using classical cryptography accumulates the same backward-looking damage profile: ciphertexts that cannot be recalled, address relationships that become permanent, proofs that become forgeable in retrospect. Private voting protocols, confidential auctions, anonymous credentials, and any application that commits secret-bearing data to a public ledger face structurally analogous constraints.

Sombra demonstrates that post-quantum private payments at genesis are not only possible but practical: a working construction on Solana, deploying as a standard program, with all the quantum attack vectors compromising classical shielded-protocol designs eliminated by primitive choice rather than future migration. The architectural question raised in our prior work admits a constructive answer. We invite the broader privacy community to extend the construction to other domains where the same backward-looking damage model applies.

References

[Arc] Arcium. Umbra: confidential computation on solana. URL: https://arcium.com.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Paper 2018/046, 2018. URL: https://eprint.iacr.org/2018/046.

[Bon26a] Bonsol Labs. The zcash dilemma, part 2: what quantum migration actually requires, April 2026.

[Bon26b] Bonsol Labs. The zcash dilemma: what quantum migration actually looks like for privacy chains, April 2026.

[Bow25] Sean Bowe. Tachyon: scaling zcash with oblivious synchronization. Cryptology ePrint Archive, Paper 2025/2031, 2025. URL: https://eprint.iacr.org/2025/2031.

[Bz26] Sean Bowe and Dev (zkDragon). Whiteboard session on zcash post-quantum roadmap, April 2026. URL: https://x.com/TachyonZcash/status/2040174642741342598.

[BZG+26] Ryan Babbush, Adam Zalcman, Craig Gidney, Michael Broughton, Tanuj Khattar, Hartmut Neven, Tommaso Bergamaschi, Justin Drake, and Dan Boneh. Securing elliptic curve cryptocurrencies against quantum vulnerabilities: resource estimates and mitigations. Google Quantum AI, March 2026. URL: https://quantumai.google/static/site-assets/downloads/cryptocurrency-whitepaper.pdf.

[NIS24a] NIST. Module-lattice-based digital signature standard (ml-dsa). FIPS 204, 2024.

[NIS24b] NIST. Module-lattice-based key-encapsulation mechanism standard (ml-kem). FIPS 203, 2024.

[Pro] Project Tachyon. Project tachyon. URL: https://tachyon.z.cash.

[Sol] Solana Foundation. Token-2022 confidential transfers. URL: https://spl.solana.com/token-2022/extensions#confidential-transfer.

[Tha25] Justin Thaler. Quantum computing and blockchains: matching urgency to actual threats. a16z crypto, December 2025. URL: https://a16zcrypto.com/posts/article/quantum-computing-misconceptions-realities-blockchains-planning-migrations/.