SNARC AXIOM: Ledgerless Privacy-Preserving Payments via Accumulators, Nullifier Roots, and Hybrid BFT+VDF Finality

Anonymous Author(s)

ABSTRACT

We present SNARC AXIOM (AXIOM), a cryptographic payment protocol that achieves double-spend prevention, full transaction privacy, and fast finality without maintaining a global transaction ledger as the canonical state repository. Existing privacy-preserving payment systems, including Zcash and its predecessors, retain a blockchain as the substrate for nullifier sets and coin existence records, imposing storage and bandwidth costs that grow linearly with transaction history—creating long-run pressure toward centralization through increasing full-node burden. AXIOM replaces the ledger with two compact state objects: a dynamic RSA accumulator over coin commitments (constant 256 bytes) and a Sparse Merkle Tree root commitment over the spent nullifier set (constant 32 bytes). Spent-set growth is captured by this constant-size root and can be checkpointed and archived without affecting verification. Transaction validity is established entirely through zero-knowledge proofs; coin existence is verified via accumulator membership witnesses; and double-spend prevention follows from the uniqueness of nullifiers derived via a pseudorandom function. Global consistency of these two state objects is achieved through a hybrid finality sublayer combining BFT threshold signatures with VDF-based ordering beacons, producing finalization certificates in place of ledger entries. We formalize six cryptographic security properties—correctness, unforgeability, value conservation, double-spend resistance, anonymity, and non-malleability—and three distributed-systems properties—safety, liveness, and atomic state transition—providing proof sketches under standard assumptions (discrete logarithm, collision resistance, PRF security, RSA strong assumption, and, where required by the SNARK extractor, a knowledge assumption standard in pairing-based proof system analyses). We further present a permissionless committee selection mechanism based on VRF sortition with stake-weighted admission, and a spam-resistance layer compatible with both semi-permissioned and fully open deployments. Concrete instantiation with Groth16 over BLS12-381, RSA-2048 accumulators, BLS threshold signatures, and ECVRF yields an estimated end-to-end finality of 2–5 seconds and approximately 60 TPS on reference hardware.

CCS CONCEPTS

• **Security and privacy** → **Cryptography**; *Distributed systems security*; • **Theory of computation** → *Interactive proof systems.*

KEYWORDS

privacy, zero-knowledge proofs, accumulators, nullifiers, Byzantine fault tolerance, verifiable delay functions, ledgerless payments, double-spend prevention

Conference'17, July 2017, Washington, DC, USA
2026. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Anonymous Author(s). 2026. SNARC AXIOM: Ledgerless Privacy-Preserving Payments via Accumulators, Nullifier Roots, and Hybrid BFT+VDF Finality. In . ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

1.1 Problem Statement

Since Bitcoin introduced the blockchain as a global append-only ledger, essentially all decentralized payment systems have inherited this architectural decision. The ledger simultaneously serves three functions: establishing a canonical transaction order, preventing double-spending by recording all past nullifiers, and providing a verifiable audit trail for coin validity. This coupling is not architecturally necessary. The three functions can be decomposed—and the audit trail, which drives ledger growth, can be eliminated entirely if coin validity is established by cryptographic proof rather than chain membership.

Existing privacy-preserving systems such as Zcash [6] demonstrate that double-spend prevention can be reduced to a nullifier set and coin validity to zero-knowledge proofs [1]. Yet Zcash retains a blockchain as the substrate for these structures, requiring every full node to store and synchronize the complete transaction history. The resulting state growth is linear in transaction count, creating long-run pressure toward centralization by increasing storage, bandwidth, and initial synchronization cost for participants.

1.2 This Work

We ask: *can a payment system achieve the security properties of Zcash—double-spend resistance, full transaction privacy, value conservation—without any global transaction ledger?*

We answer affirmatively, subject to standard cryptographic hardness assumptions and a partial synchrony network model [5]. Our construction, SNARC AXIOM (AXIOM for short), maintains only two global state objects whose sizes are independent of transaction history:

- **ACC**: a dynamic RSA accumulator over coin commitments—constant 256 bytes regardless of coin set size; and
- **SpentSetRoot**: a Sparse Merkle Tree root over nullifier hashes—constant 32 bytes, with the underlying set checkpointed and archived separately.

These objects are finalized by a BFT committee producing threshold signatures over batched state transitions, yielding finalization certificates that replace ledger entries. A VDF beacon [14, 15] constrains leader discretion in transaction ordering without requiring a trusted clock.

1.3 Contributions

- **C1 — Ledgerless payment protocol.** We define SNARC AXIOM and prove that it achieves double-spend resistance, value conservation, and full transaction anonymity without a global transaction ledger as the canonical state repository, relying instead on compact state objects and finalization certificates.

- **C2 — Formal security model.** We provide a complete threat model, six cryptographic security definitions, and proof sketches under standard assumptions: discrete logarithm hardness, collision resistance, PRF security, RSA strong assumption, and SNARK knowledge soundness [8, 12].
- **C3 — Atomic state transition with BFT finality.** We define a deterministic State Transition Function $\mathsf{Apply}(S, B) \to S'$ (Section 6) and prove it is atomic under the hybrid BFT+VDF consistency sublayer [3], replacing the blockchain's role in ordering and finalizing state updates. The threshold signature covers both the batch content and the resulting state roots jointly, making partial observation impossible at the protocol level.
- **C4 — Permissionless committee selection.** We formalize VRF-based sortition [9, 10] with stake-weighted admission (Section 9), proving sybil non-amplification, expected committee size bounds, and compatibility with the BFT safety threshold.
- **C5 — Spam and DoS resistance.** We define a two-tier mempool architecture with three composable admission mechanisms—stake tickets, PoW-lite hashcash, and VRF tickets—and prove bounded verification work and spam cost dominance (Section 7).
- **C6 — Concrete instantiation and benchmarks.** We instantiate Axiom with Groth16 over BLS12-381, RSA-2048 accumulator, BLS threshold signatures, and ECVRF (RFC 9381), providing circuit constraint estimates, latency decomposition, throughput analysis, and a phased implementation roadmap (Section 10).

1.4 Comparison to Related Work

Axiom's cryptographic core shares its nullifier-and-ZK-proof structure with Zerocash [1] and Zcash [6]. The key architectural departure is the elimination of the blockchain as state substrate: Zcash uses the chain as the nullifier set carrier and coin existence log, while Axiom replaces both with ACC and SpentSetRoot finalized by BFT threshold certificates. This direction parallels Utreexo [4], which replaces Bitcoin's UTXO chain history with a compact accumulator—but applied to a privacy-preserving, ledgerless setting. The BFT finality and partial synchrony model follows Castro–Liskov [3] and Dwork–Lynch–Stockmeyer [5]. Permissionless committee selection extends Algorand's VRF sortition [9] with adaptive sizing and epoch rotation. VDF-based ordering follows Pietrzak [14] and Wesolowski [15]; for a unified treatment see Boneh et al. [2].

1.5 Paper Organization

Sections 5–10 present the protocol and analysis in order: the hybrid BFT+VDF consistency sublayer, accumulator atomicity, spam resistance, epoch rotation, permissionless committee selection, and concrete benchmarks. Section 12 surveys related work and Section 13 concludes.

2 NOTATION AND PRELIMINARIES

2.1 Notation

Let λ denote the security parameter. Let $\mathbb{G}$ be a cyclic group of prime order q with generators g, h such that $\log_g h$ is unknown. We write $x \xleftarrow{\$} X$ for uniform sampling from set X, and $\mathsf{negl}(\lambda)$ for a negligible function in λ. Ordered lists are written $[x_i]_{i=1}^{m}$; $\|$ denotes bitstring concatenation.

The global state at finalization step k is:

$$S_k = (\mathsf{ACC}_k,\ \mathsf{Root}_k,\ epoch_k,\ seq_k),$$

where ACC_k is the accumulator value, Root_k is the Sparse Merkle Tree (SMT) root committing to spent nullifiers, and $(epoch_k, seq_k)$ are monotonically increasing counters.

2.2 Pedersen Commitments

Over group $\mathbb{G}$, define:

$$\mathsf{Com}(m; r) = g^m \cdot h^r.$$

Pedersen commitments are perfectly hiding and computationally binding under the discrete logarithm (DL) assumption in $\mathbb{G}$. In Axiom, a coin commitment encodes both the value and a secret seed:

$$cm = \mathsf{Com}(\langle v, \rho \rangle; r),$$

where v is the coin value, ρ is a secret serial seed, and r is commitment randomness.

2.3 Pseudorandom Function

$\mathsf{PRF}_s(\cdot)$ is a secure PRF keyed by spend key s. Outside the ZK circuit we instantiate it with BLAKE3; inside the circuit we use Poseidon [11] for efficient constraint generation.

2.4 RSA Accumulator

Let $N = pq$ be an RSA modulus with unknown factorization and generator $g \in \mathbb{Z}_N^*$. Let $H' : \{0,1\}^* \to \mathbb{P}$ be a deterministic hash-to-prime function (smallest prime p such that $p = H(\cdot \| ctr)$, iterated over a counter; average 3–4 iterations by Bertrand's postulate).

The accumulator over a set $\{x_i\}$ is:

$$\mathsf{ACC} = g^{\prod_i H'(x_i)} \bmod N.$$

Batch update for a list $[x_i]_{i=1}^{m}$:

$$\mathsf{ACC}' = \mathsf{ACC}^{\prod_{i=1}^{m} H'(x_i)} \bmod N.$$

A membership witness w for element x satisfies:

$$w^{H'(x)} \equiv \mathsf{ACC} \pmod N.$$

Security relies on the strong RSA assumption.

2.5 Sparse Merkle Tree

SMT is a Sparse Merkle Tree over 2^λ leaves (256-bit address space). Root_k commits to the set of spent nullifiers (stored as $H(sn)$ at leaf $H(sn)$). Non-membership proofs follow directly from the tree structure under collision resistance of the hash function (Poseidon/BLAKE3 as appropriate).

2.6 Threshold Signature Scheme

TSig = (KeyGen, SignShare, Combine, Verify) is a (t, n)-threshold signature scheme. In epoch e we set $t_e = \lfloor 2n_e/3 \rfloor + 1$, tolerating up to $f_e < n_e/3$ Byzantine validators. We instantiate with BLS threshold signatures over BLS12-381: aggregated signature size is 48 bytes (independent of n).

2.7 VDF Beacon

A Verifiable Delay Function [14, 15] VDF = (Setup, Eval, Verify) satisfies sequentiality ($\mathsf{Eval}(x, T)$ requires $\approx T$ sequential steps), uniqueness, and efficient verification ($O(\log T)$). The epoch beacon derives:

$$seed_{e+1} = H\big(seed_e \parallel \mathsf{VDF}(seed_e, T_{\mathrm{vdf}}) \parallel H(B_{\mathrm{last},e})\big),$$

producing an unpredictable, publicly verifiable ordering key for each epoch.

2.8 NIZK Proof System

We assume a succinct NIZK Π = (Setup, Prove, Verify) satisfying completeness, knowledge soundness (via an extractor), and zero-knowledge. The v1 instantiation uses Groth16 [12] over BLS12-381. Where non-malleability is required (Section 7), we additionally require simulation-extractability, achievable via PLONK [8] or Fiat–Shamir with appropriate assumptions.

Trusted setup note. Groth16 requires a circuit-specific trusted setup ("toxic waste" must be discarded). A multi-party computation ceremony mitigates this; eliminating the requirement entirely calls for a setup-free system (e.g., STARKs) and is deferred to future work (Section 13).

3 THREAT MODEL

3.1 Adversarial Capabilities

We consider a PPT adversary $\mathcal{A}$ that:

- controls an arbitrary fraction of non-validator network nodes;
- schedules and delays messages arbitrarily before GST (the Global Stabilization Time of the DLS partial-synchrony model [5]);
- adaptively corrupts up to $f_e < n_e/3$ validators within each epoch e; and
- is computationally bounded (PPT).

$\mathcal{A}$ may attempt to: forge coin proofs, double-spend a coin, inflate value, deanonymize senders, or violate state consistency during epoch transitions. $\mathcal{A}$ may also attempt sustained spam/DoS (Section 7).

3.2 Cryptographic Assumptions

The following hardness assumptions are used throughout; each is stated where it is first required.

- **DL**: Discrete logarithm in $\mathbb{G}$ is hard.
- **CRH**: H is collision resistant.
- **PRF**: $\mathsf{PRF}_s(\cdot)$ is a secure pseudorandom function.
- **sRSA**: Strong RSA assumption in $\mathbb{Z}_N^*$ (underlies accumulator security).
- **KS**: Knowledge soundness of the SNARK proof system (Groth16: holds under the knowledge-of-exponent assumption, which is non-falsifiable but standard in pairing-based SNARK analyses; PLONK: holds in the algebraic group model).
- **ZK**: Zero-knowledge of the SNARK proof system.
- **TSig-UF**: Threshold signature unforgeability (BLS TSig under DL).
- **VDF-Seq**: Sequentiality of the VDF (modular squaring in a group of unknown order).

3.3 Network Model

We use the partial synchrony model of Dwork–Lynch–Stockmeyer [5]: after an unknown GST, all messages arrive within a known bound Δ. Before GST, *safety* is maintained; *liveness* is guaranteed only after GST.

3.4 Validator Corruption Bound

In epoch e, there are n_e validators; at most $f_e < n_e/3$ may be Byzantine. Finality threshold: $t_e = \lfloor 2n_e/3 \rfloor + 1$. Byzantine validators may send conflicting messages, withhold signatures, or attempt to equivocate; honest validators follow the protocol exactly.

3.5 Security Goals

We target the following properties, proved in the sections indicated.

Correctness (Section 5): Honest execution produces accepted transactions.
Unforgeability (Section 6): No PPT adversary can spend a coin without knowing its witness.
Value Conservation : No PPT adversary can inflate coin values.
Double-Spend Resistance (Section 5): The same nullifier cannot be finalized twice.
Anonymity (Section 9): An adversary cannot distinguish which of two candidate coins produced a challenge transaction.
Non-Malleability : A valid transaction cannot be transformed to steal value without knowing witness material.
Safety (Section 5): No two honest verifiers accept conflicting finalized state.
Liveness (Section 5): After GST, valid transactions are finalized within $O(T_{\mathrm{beacon}} + \kappa\Delta)$.
Atomic Transition (Section 6): ACC and SpentSetRoot update jointly or not at all.

3.6 Out of Scope

We explicitly do not claim:

- *Full censorship resistance*: we target only weak fairness via VDF ordering and attributable misbehavior (Section 5).
- *Bitcoin-equivalent trustlessness*: Groth16 requires a trusted setup; the RSA accumulator requires a trusted modulus generation. Eliminating both requires setup-free proofs and class-group accumulators, which we defer to future work.

- *Network-layer DoS immunity*: a state-level adversary with unlimited bandwidth can flood the physical network; this is outside the scope of this protocol.

4 PROTOCOL OVERVIEW AND TRANSACTION VALIDITY

4.1 Coins as Cryptographic Objects

A coin is not a ledger entry. It is a commitment paired with a private witness:

$$\mathsf{Coin} \;:=\; (cm;\ v, \rho, r, s, w),$$

where $cm = \mathsf{Com}(\langle v, \rho\rangle; r)$ is the public commitment, v is the coin value, ρ is a secret serial seed, r is commitment randomness, s is the spend key, and w is an RSA accumulator membership witness for cm in the current ACC. The global state does not track balances per identity; ownership is demonstrated by proving knowledge of a witness consistent with the current compact state.

4.2 Nullifiers and Double-Spend Prevention

For each coin define its *nullifier*:

$$sn = \mathsf{PRF}_s(\rho).$$

A nullifier is revealed only at spend time. The spent nullifier set is committed by Root_k. Double-spending reduces to ensuring the same sn cannot be accepted under two distinct finalized states (proved in Theorems 5.5 and 8.2).

4.3 Transaction Types

SpendTx (primary). $TX = (sn,\ cm_{new},\ \pi)$, where π is a SNARK proof for relation $\mathcal{R}$ (Section 4.5).

MintTx (issuance). $MintTx = (cm_{new},\ v_{pub},\ \sigma_{authority})$, where the issuing authority signs the committed value. The new coin is inserted into ACC without a nullifier reveal. Full analysis of issuance policy is deferred to the application layer.

BurnTx (redemption). $BurnTx = (sn,\ v_{pub},\ \pi_{burn})$, where π_{burn} proves knowledge of a valid coin with value v_{pub} and correctly derived nullifier, without creating a new commitment. The nullifier is inserted into Root; no cm_{new} is added to ACC.

The v1 security analysis (Section 5–Section 9) focuses on SpendTx; MintTx and BurnTx satisfy analogous correctness and soundness properties by construction.

4.4 Global State and Finalization Certificates

The system maintains only two compact roots:

$$S_k = (\mathsf{ACC}_k,\ \mathsf{Root}_k,\ epoch_k,\ seq_k).$$

Each finalized batch produces a certificate:

$$C_k = (B_k,\ S_k,\ \sigma_k),$$

where $\sigma_k = \mathsf{TSig}_{t_e}(H(B_k \parallel S_k.\mathsf{ACC} \parallel S_k.\mathsf{Root} \parallel \cdots))$ covers both the batch content and the resulting state roots jointly (Theorem 6.6). Clients treat C_k as the canonical finality artifact, replacing ledger entries.

4.5 The ZK Relation $\mathcal{R}$

Public inputs. $(\mathsf{ACC},\ \mathsf{Root},\ sn,\ cm_{new})$

Witness. $(v, \rho, r, s, w, \rho_{new}, r_{new})$

Constraints. The prover demonstrates:

1. *Old commitment well-formed:* $cm = \mathsf{Com}(\langle v, \rho\rangle;\ r)$.
2. *Membership in accumulator:* $\mathsf{VerifyMem}(\mathsf{ACC},\ cm,\ w) = 1$.
3. *Nullifier correctness:* $sn = \mathsf{PRF}_s(\rho)$.
4. *Value conservation:* $cm_{new} = \mathsf{Com}(\langle v, \rho_{new}\rangle;\ r_{new})$. For multi-output extensions, the circuit enforces $\sum_j v_{out,j} = v_{in}$.
5. *Spend authorization:* The prover knows spend key s bound to the committed coin. In the circuit this is enforced by deriving a key commitment $\mathsf{Com}(s;\ r_s)$ embedded in the old coin's committed payload, ensuring only the holder of s can produce a satisfying witness.

The *unspent condition* ($H(sn) \notin \mathsf{Root}$) is verified outside the circuit against the current state snapshot, keeping the circuit small and enabling cheap Tier-0 filtering (Section 7).

Proof and verification.

$$\pi \;\leftarrow\; \mathsf{Prove}_{\mathcal{R}}(\mathsf{ACC},\ \mathsf{Root},\ sn,\ cm_{new};\ witness)\,.$$

Acceptance at the protocol level requires: (i) $\mathsf{ZKVerify}(\pi, \mathsf{ACC}, sn, cm_{new}) = 1$; (ii) $\mathsf{NonMember}(\mathsf{Root}, H(sn)) = 1$; (iii) batch-level nullifier distinctness; and (iv) threshold signature consistency of the finalization certificate.

4.6 High-Level Acceptance Rule

Given finalized state S_{k-1} and candidate TX:

1. Verify π against ACC_{k-1} and public inputs.
2. Verify $H(sn) \notin \mathsf{Root}_{k-1}$.
3. Include TX in a BFT-proposed batch.
4. Upon threshold finalization, produce C_k with atomic state update (Section 6).

5 STATE CONSISTENCY WITHOUT A LEDGER

5.1 Overview and Design Goals

Axiom's cryptographic core requires no ledger; however, preventing double-spending requires that two global state objects remain consistent on a single history:

- **SpentSet**: the append-only set of spent nullifiers; and
- **ACC**: the accumulator state over valid coin commitments.

This section achieves the following guarantees without a global blockchain:

- **G1 — Safety.** The same nullifier sn cannot be finalized twice.
- **G2 — Atomicity.** When a transaction is finalized, the sn and *cm_new* updates are jointly final.
- **G3 — Liveness.** Valid transactions are eventually finalized once the network stabilizes.
- **G4 — Weak Fairness.** No single actor can indefinitely censor a specific transaction.

The hybrid design uses (A) a BFT Threshold Finality Layer for short-term ordering, atomicity, and finality, and (B) a VDF Time-Ordering Beacon for long-term ordering fairness and anti-censorship.

5.2 System Model

Definition 5.1 (Partially Synchronous Network [5]). *There exists an unknown* Global Stabilization Time *(GST) such that all messages sent after GST arrive within known bound* Δ.

Definition 5.2 (Byzantine Validator Set). *Of n validators, at most $f < n/3$ are Byzantine (arbitrarily malicious). The BFT threshold is $t = \lfloor 2n/3 \rfloor + 1$.*

Definition 5.3 (Threshold Signature Scheme). *A (t, n)-TSS* $= (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Combine}, \mathsf{Verify})$ *produces a valid aggregated signature if and only if at least t honest shares are combined.*

5.3 VDF Time-Ordering Beacon

Definition 5.4 (VDF [14, 15]). $\mathsf{VDF} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Verify})$ *satisfies:*

- Sequentiality*:* $\mathsf{Eval}(x, T) \to (y, \pi)$ *requires exactly T sequential steps.*
- Uniqueness*: each x yields a unique valid y.*
- Efficient verification*:* $\mathsf{Verify}(x, y, \pi, T) = 1$ *in $O(\log T)$.*

Each epoch's VDF beacon seed is:

$$seed_{e+1} = H\big(seed_e \parallel \mathsf{VDF}(seed_e, T_{\text{vdf}}) \parallel H(B_{\text{last},e})\big)\,.$$

Every transaction receives a deterministic priority key $prio(TX) = H(sn \| cm\_new \| y_e)$, constraining leader discretion without a trusted clock.

5.4 BFT Threshold Finality Protocol

Each epoch, a leader proposes a batch:

$$B_k = (epoch, k, prev\_hash, tx\_list, y_e, \pi_{\text{vdf}}).$$

Validators (i) verify the VDF output, (ii) verify each ZK proof in tx_list, and (iii) check nullifier non-membership against $S_{k-1}.\mathsf{SpentSetRoot}$. Upon collecting $\sigma_k = \mathsf{TSig}_t(H(B_k))$, the batch is final and the state is updated atomically (Section 6).

5.5 Safety and Liveness

Theorem 5.5 (Safety). *With $f < n/3$ Byzantine validators and a correct BFT protocol, no two distinct accepted batches can contain transactions sharing the same nullifier sn.*

Proof sketch. Two conflicting batches each require t signatures. Any two quorums of size t over $n = 3f + 1$ validators share at least $f + 1$ nodes; at least one is honest. An honest validator refuses to sign a batch containing a nullifier already present in the committed SpentSetRoot, or a duplicate within the batch. Hence two conflicting finals are impossible. □

Theorem 5.6 (Liveness). *After GST, every valid transaction submitted by an honest sender is finalized within $O(T_{\text{beacon}} + \kappa\Delta)$ time, where κ is a small BFT-protocol constant.*

Proof sketch. After GST, messages arrive within Δ; BFT protocols of the HotStuff class [3] terminate in $O(\Delta)$ rounds under an eventually honest leader. VDF evaluation completes in T_{beacon} before epoch end. □

5.6 Weak Fairness via VDF Ordering

Validators soft-enforce that the leader's batch ordering respects $prio(\cdot)$. Systematic deviation constitutes *provable misbehavior*: at least $f + 1$ honest validators observe the violation and can initiate a slashing or reputation penalty. This does not guarantee full censorship-resistance, but raises the cost of sustained censorship to the level of observable, attributable protocol violation.

6 ACC UPDATE ATOMICITY

6.1 Motivation

Section 5 guarantees that the same nullifier cannot be finalized twice. A complementary guarantee is required: when a batch is finalized, both the accumulator update and the nullifier insertion succeed together, or neither takes effect. This section formalizes that guarantee.

6.2 State and Transition Function

Definition 6.1 (Global State).

$$S = (\mathsf{ACC},\ \mathsf{SpentSetRoot},\ epoch,\ seq)$$

where $\mathsf{ACC} \in \mathbb{Z}_N^*$ *is the RSA accumulator value and* $\mathsf{SpentSetRoot} \in \{0,1\}^\lambda$ *is the Sparse Merkle Tree root over hashed nullifiers. The genesis state is $S_0 = (g, \bot, 0, 0)$.*

Definition 6.2 (Batch).

$$B_k = (epoch,\ k,\ prev\_hash,\ tx\_list,\ \sigma)$$

where $tx\_list = [TX_1, \dots, TX_m]$, $TX_i = (sn_i, cm\_new_i, \pi_i)$, and $\sigma = \mathsf{TSig}(H(B_k \setminus \sigma))$.

Definition 6.3 (State Transition Function). $\mathsf{Apply}(S, B_k) = S'$ *if and only if all six preconditions hold:*

1. $B_k.prev\_hash = H(S)$.
2. $B_k.seq = S.seq + 1$.
3. $\mathsf{TSig.Verify}(vk,\ H(B_k \setminus \sigma),\ \sigma) = 1$.
4. $\forall i:\ \mathsf{ZKVerify}(\pi_i, S.\mathsf{ACC},\ sn_i,\ cm\_new_i) = 1$.
5. $\forall i \neq j:\ sn_i \neq sn_j$ *(no intra-batch duplicate).*
6. $\forall i:\ \mathsf{NonMember}(S.\mathsf{SpentSetRoot},\ H(sn_i)) = 1$.

If all hold, the output state is:

$$\mathsf{ACC}' := \mathsf{ACC}^{\prod_i H'(cm\_new_i)} \bmod N,$$

$$\mathsf{SpentSetRoot}' := \mathsf{SMT.BatchInsert}(\mathsf{SpentSetRoot},\ [H(sn_i)]_i),$$

$$S' := (\mathsf{ACC}',\ \mathsf{SpentSetRoot}',\ epoch,\ seq + 1).$$

If any precondition fails, Apply *is undefined and S is unchanged.*

Here $H' : \{0,1\}^* \to \mathbb{P}$ is a collision-resistant hash-to-prime function (deterministic probable-prime search, average 3–4 iterations via Bertrand's postulate).

6.3 Atomicity and Determinism

Definition 6.4 (Atomicity). $\mathsf{Apply}(S, B)$ is atomic *if: either all preconditions hold and both* ACC' *and* $\mathsf{SpentSetRoot}'$ *are jointly produced as S', or B is entirely rejected and S is unchanged. No intermediate state is observable.*

Lemma 6.5 (Determinism). Apply *is a pure function: the same (S, B) always yields the same S' or always fails.*

Proof. Every sub-operation—H, TSig.Verify, ZKVerify, modular exponentiation mod N, and SMT insertion ordered by *tx_list* index—is deterministic and free of external state. □

Theorem 6.6 (Atomic State Transition). *If* Apply(S, B) *is accepted, then for all* i:

$$cm\_new_i \in \mathrm{Acc}(S'.\mathrm{ACC}) \quad and \quad sn_i \in \mathrm{SpentSet}(S'.\mathrm{SpentSetRoot}),$$

and both memberships are certified by the same threshold signature $\sigma' = \mathrm{TSig}(H(B \| S'.\mathrm{ACC} \| S'.\mathrm{SpentSetRoot} \| \cdots))$.

Proof. The signature preimage includes ACC′ and SpentSetRoot′ jointly. TSig unforgeability (under discrete log) prevents a valid σ' in which either field is modified independently. A client accepting σ' accepts both fields simultaneously. □

Corollary 6.7 (Global State Consistency). *All honest verifiers computing* $S_0, S_1, S_2, \ldots$ *independently via* Apply *arrive at identical sequences.*

Proof. By induction: S_0 is fixed; if S_{k-1} is identical across honest verifiers and B_k is identical (BFT total order), then Theorem 6.5 yields identical S_k. □

6.4 Light Client Verification

Corollary 6.8 (Light Client Protocol). *A coin owner with private witness* (v, ρ, r, s, cm) *verifies coin liveness without the full state as follows:*

1. *Compute* $sn_{coin} = \mathrm{PRF}_s(\rho)$ *locally (never revealed).*
2. *Obtain the latest finalized* $C_k = (B_k, S_k, \sigma_k)$ *and verify* $\mathrm{TSig.Verify}(vk, \cdot, \sigma_k) = 1$.
3. *Check* $\mathrm{NonMember}(S_k.\mathrm{SpentSetRoot},\ H(sn_{coin})) = 1$.
4. *Check* $\mathrm{Member}(S_k.\mathrm{ACC},\ cm,\ w) = 1$.

Cost: $O(\log n)$ *per Merkle proof,* $O(1)$ *for TSig and accumulator verification.*

7 SPAM AND DOS RESISTANCE

7.1 Motivation

Axiom's cryptographic core guarantees transaction correctness but imposes no economic friction on transaction *submission*. Without such friction, a PPT adversary may mount: mempool flooding, ZK-verify exhaustion, state bloat, or censorship-by-congestion. This section defines an anti-spam sublayer that is composable with both semi-permissioned and fully permissionless admission models.

7.2 Design Principles

- **P1 (Early Reject).** The most expensive check (ZK verify) is performed last.
- **P2 (Sender Pays).** The party generating network load bears its cost.
- **P3 (Bounded Admission).** Per-epoch transaction count has a deterministic upper bound $M_{\max}$.
- **P4 (Punish Proven Abuse).** Provable spam triggers stake-credit burning or admission banning.
- **P5 (Composable).** Mechanisms work under both stake-based and fully open admission.

7.3 Two-Tier Mempool

Incoming envelopes $TX^* = (TX, fee, anti\_replay, admission\_proof)$ are processed in two tiers.

Tier 0 (Cheap). In order: (1) size and format bounds; (2) admission proof verification; (3) fee floor $fee \geq f_{\min}(epoch)$; (4) nullifier shape check; (5) duplicate suppression via (sn, cm_new) hash cache. Failure at any step causes immediate drop—no ZK verification is invoked.

Tier 1 (Expensive). Only for Tier 0 survivors: (6) $\mathrm{ZKVerify}(\pi, \mathrm{ACC}, sn, cm\_new)$; (7) $\mathrm{NonMember}(\mathrm{SpentSetRoot}, H(sn))$; (8) intra-batch conflict check (leader).

7.4 Admission Mechanisms

Three composable mechanisms are available; deployments choose one or more.

Definition 7.1 (Stake Ticket (Mechanism A)). *A user depositing* $stake_u$ *tokens receives a per-epoch quota:*

$$quota_u(e) = \lfloor stake_u / stake\_unit \rfloor.$$

Each submitted TX^* *consumes one ticket* $ticket_{id} = H(pk_u \| e \| ctr)$. *Double-use is detected via a per-epoch SMT ticket registry.*

Definition 7.2 (PoW-lite / Hashcash (Mechanism B)). *For permissionless entry, the sender finds a nonce such that:*

$$H(e \| TX \| nonce) < 2^{-d(e)},$$

where $d(e)$ *is the epoch difficulty parameter, adjusted dynamically based on mempool pressure. Verification cost: one hash evaluation. Expected production cost:* $2^{d(e)}$ *hash evaluations.*

Definition 7.3 (VRF Ticket (Mechanism C)). *The sender evaluates:*

$$ticket = \mathrm{VRF.Eval}(sk,\ seed_e \| H(TX)),$$

and is admitted if $H(ticket) < \tau_{tx}(e)$. *This provides probabilistic admission without wasting CPU, and integrates naturally with the epoch sortition beacon (Section 9).*

7.5 Fee Market and Batch Selection

The leader selects admitted transactions by:

$$score(TX^*) = fee / bytes,$$

with ties broken by the VDF priority key $prio(TX)$. Batch caps $M_{\max}(e)$ and $B_{\max}(e)$ are enforced deterministically.

7.6 Formal Properties

Theorem 7.4 (Bounded Verification Work). *Let* R_0 *be the Tier 0 pass rate. The expected number of expensive (Tier 1) verifications per unit time is at most* $R_0 \cdot CAP_{\max}$, *and* R_0 *is upper-bounded by the adversary's resource expenditure under any active admission mechanism.*

Theorem 7.5 (Spam Cost Dominance). *For parameters* $(stake\_unit, d(e), \tau_{tx}(e),$ *chosen such that the expected cost of passing Tier 0 exceeds* $\alpha \cdot C_{verify}$ *for some* $\alpha > 1$, *sustained Tier 1-targeting spam is economically dominated.*

Theorem 7.6 (Batch Bloat Resistance). *Per-epoch state growth satisfies:*

$$\Delta|\mathsf{SpentSet}| \leq M_{\max}, \quad \Delta\mathsf{ACC\_updates} \leq M_{\max}.$$

State growth rate is a controlled system parameter, not adversary-controlled.

7.7 Honest Disclosure

DoS is made *costly*, not impossible. A nation-state-level bandwidth adversary can flood the network layer; this requires out-of-scope network-level mitigations. Fee token economics and VRF sybil resistance are deployment concerns not fully specified here.

8 EPOCH ROTATION AND COMMITTEE FRESHNESS

8.1 Motivation

Section 5 assumed a fixed validator set V. In practice, validators join or leave between epochs, keys must be rotated to maintain forward secrecy, and a static committee creates long-range attack exposure. This section formalizes epoch transitions and proves that safety and liveness are preserved across committee changes.

8.2 Epoch and Validator Set Model

Definition 8.1 (Epoch).

$$\mathsf{Epoch}_e = (V_e,\ vk_e,\ T^e_{\text{start}},\ T^e_{\text{end}},\ seed_e),$$

where V_e is the validator set, vk_e the aggregated threshold verification key, and $seed_e$ the VDF beacon seed. The BFT parameters are $n_e = |V_e|$, $f_e < n_e/3$, and $t_e = \lfloor 2n_e/3 \rfloor + 1$.

8.3 Handoff MPC and Key Rotation

Key rotation proceeds in three in-epoch phases.

Phase 1: DKG (first two-thirds of epoch e). V_{e+1} runs a Distributed Key Generation protocol (e.g., Feldman VSS [7]) producing secret shares distributed among V_{e+1} and public key vk_{e+1}.

Phase 2: Handoff Certificate (last third of epoch e).

$$HC_e = \mathsf{TSig}_{V_e}\left(H\left(vk_{e+1} \parallel e{+}1 \parallel S_{\text{last},e}\right)\right). \tag{1}$$

HC_e certifies vk_{e+1} and binds it to the last finalized state of epoch e.

Phase 3: Epoch transition. The first batch of epoch e+1 includes HC_e. Clients verify HC_e before accepting signatures under vk_{e+1}. Members of V_e delete their shares of sk_e (best-effort forward secrecy).

8.4 Safety Across Rotation

Theorem 8.2 (Epoch Transition Safety). *Under Theorem 5.5's assumptions applied to both V_e and V_{e+1}, the same nullifier sn cannot be finalized in two different epochs.*

Proof sketch. A transaction finalized in epoch e inserts sn into $S_{\text{last},e}$.SpentSetRoot. The first batch of epoch e+1 chains to $H(S_{\text{last},e})$ (via *prev_hash*) and V_{e+1} checks non-membership against $S_{\text{last},e}$.SpentSetRoot. An adversary cannot present a forged $S_{\text{last},e}$ because HC_e is bound to the genuine last state and is unforgeable under TSig unforgeability (Section 5). □

8.5 Liveness Across Rotation

Theorem 8.3 (Epoch Transition Liveness). *If DKG completes and HC_e is produced within epoch e, then epoch e+1 begins with a fully operational committee within $O(\Delta)$ after T^e_{end}.*

8.6 Long-Range Attack Resistance

Definition 8.4 (Weak Subjectivity Window). *Clients do not accept state assertions older than W epochs without an out-of-band trusted checkpoint.*

Theorem 8.5 (Long-Range Resistance). *Rewriting state from epoch $e < current - W$ requires one of: (i) forging HC_{e-1} (ruled out by TSig unforgeability), or (ii) solving T_{beacon} sequential VDF squarings per epoch (sequential hardness of VDF), or (iii) presenting a forged checkpoint (ruled out by the weak subjectivity assumption).*

8.7 Validator Churn Tolerance

Definition 8.6 (Churn Rate). *$churn_e = |V_e \triangle V_{e+1}|/n$.*

Theorem 8.7 (Churn Safety Bound). *If $|V_e \cap V_{e+1}|$ contains at least t_e honest validators, then safety is preserved across the epoch boundary.*

Corollary 8.8. *Maximum safe churn per epoch is $churn_e \leq 1/3$. Full committee replacement in a single epoch transition is unsafe.*

9 PERMISSIONLESS COMMITTEE SELECTION

9.1 Motivation

Section 8 introduced epoch-based committee rotation under a stake-weighted admission model. This section extends that model to fully permissionless participation via VRF-based sortition, proving sybil resistance and compatibility with the BFT safety threshold.

9.2 Design Goals

- **G9 (Open Entry).** Any node depositing minimum stake may participate.
- **G10 (Unpredictability).** Committee membership for epoch e cannot be predicted before epoch $e-1$ ends.
- **G11 (Proportionality).** Selection probability is proportional to stake.
- **G12 (Sybil Resistance).** Splitting stake across k identities yields no additional advantage.

9.3 Beacon Seed Chain

Definition 9.1 (Beacon Seed Chain).

$$seed_0 = H(\text{“AXIOM\_GENESIS”} \| pp), \quad seed_{e+1} = H\big(seed_e \parallel \mathsf{VDF}(seed_e, T_{\text{vdf}}) \parallel H($$

Lemma 9.2 (Seed Unpredictability). *Under VRF pseudorandomness and VDF sequentiality, the advantage of any PPT adversary controlling $f < n/3$ validators in predicting $seed_{e+1}$ at the start of epoch e is negligible.*

Proof sketch. $\mathsf{VDF}(seed_e, T_{\text{vdf}})$ requires T_{vdf} sequential steps and cannot be precomputed; honest epoch activity contributes fresh entropy via $H(B_{\text{last},e})$. By VRF pseudorandomness, knowing $seed_e$ does not allow predicting which nodes will produce high-ranking VRF outputs. □

9.4 Sortition Protocol

Definition 9.3 (Sortition). *In epoch e, each node v with stake $stake_v$ computes:*

$$(y_v, \pi_v) = \mathsf{VRF.Eval}(sk_v,\ seed_e \| \text{"COMMITTEE"} \| e).$$

Node v is selected iff:

$$\frac{H(y_v)}{2^\lambda} \leq \tau(e) \cdot \frac{stake_v}{stake\_total_e}.$$

Selected nodes broadcast $(pk_v, y_v, \pi_v, stake_v, w_{stake})$, where w_{stake} is a Merkle proof of stake.

Theorem 9.4 (Expected Committee Size). $\mathbb{E}[|V_e|] = \tau(e) \cdot (stake\_active/stake\_unit)$. *Setting* $\tau(e) = n_{target} \cdot stake\_unit/stake\_active$ *yields* $\mathbb{E}[|V_e|] = n_{target}$.

Theorem 9.5 (Sybil Non-Amplification). *An adversary with total stake S_{adv} splitting across k identities has the same expected committee seats as a single identity with stake S_{adv}.*

Proof. For k identities each with stake S_{adv}/k:

$$\Pr[\text{at least one selected}] = 1 - \prod_{i=1}^{k}\left(1 - \tau \cdot \frac{S_{adv}/k}{stake\_total}\right) \xrightarrow[k\to\infty]{} \tau \cdot \frac{S_{adv}}{stake\_total},$$

which equals the single-identity selection probability. □

9.5 Adaptive Committee Sizing

Definition 9.6 (Adaptive τ).

$$\tau(e+1) = \tau(e) \cdot \frac{n_{target}}{|V_e|}, \quad \tau(e) \geq \tau_{\min} = \frac{(3f_{target}+1) \cdot stake\_unit}{stake\_active}.$$

$\tau_{\min}$ ensures $\mathbb{E}[|V_e|] \geq 3f_{target}+1$ at all times.

9.6 Bootstrap and Genesis Committee

The first epoch uses a pre-announced genesis committee of $k \geq 3f+1$ early stake depositors. This is not a trusted setup: genesis members receive no special protocol privileges beyond epoch 0. From epoch 1 onward, sortition (Theorem 9.3) fully controls committee membership.

9.7 Honest Disclosure

Stake concentration creates plutocracy risk; mitigation options (quadratic weighting, per-validator stake caps) are deployment choices not specified here. "Nothing-at-stake" behavior (signing multiple forks) is deterred by slashing (Section 7, P4). Low overall stake participation can compress committee size toward $\tau_{\min}$, reducing liveness margin.

10 CONCRETE INSTANTIATIONS AND BENCHMARKS

10.1 Component Choices

ZK proof system. We instantiate with Groth16 [12] over BLS12-381 as the default (smallest proof size, fastest verification). PLONK [8] is the recommended upgrade path (universal trusted setup, circuit-agnostic). STARKs are noted for post-quantum scenarios at the cost of larger proofs (~100 KB).

Table 1: Approximate R1CS constraint counts for relation $\mathcal{R}$.

Constraint source	R1CS constraints
Pedersen commitment (×2)	500
Poseidon hash (×3)	1,200
PRF / BLAKE3 (Poseidon substitute)	800
RSA membership witness verify	15,000
Value range proof	500
Total	**≈ 18,000**

Accumulator. RSA-2048 accumulator with batch witness updates (Boneh et al. 201 H' maps to primes via deterministic probable-prime search. Alternative: class-group accumulator (no trusted setup; 3–5× slower).

Commitment scheme. Pedersen commitments over BLS12-381 $\mathbb{G}_1$: $\mathsf{Com}(v; r) = v \cdot G + r \cdot H$.

PRF and hash. $\mathsf{PRF}_s(\rho) = \mathsf{BLAKE3}(s\|\rho)$ (circuit-external); Poseidon hash inside ZK circuits for efficient constraint generation.

Threshold signature. BLS threshold signatures over BLS12-381. Aggregated signature size: 48 bytes (constant in n). Verification: 2 pairings ≈ 3 ms.

VDF.. Wesolowski VDF [15] with 2048-bit RSA modular squaring. T = 4,000 squarings ≈ 2 seconds on a single CPU core. Verification ≈ 5 ms.

VRF.. ECVRF over Ed25519 per RFC 9381 [10].

10.2 ZK Circuit Estimate

Estimated proving times (Groth16, 18K R1CS): RTX 4090: ≈ 400 ms; RTX 5090: ≈ 200 ms; Server FPGA: ≈ 50 ms.

10.3 End-to-End Latency

$$T_{total} = \underbrace{T_{prove}}_{200\text{–}400\text{ ms}} + \underbrace{T_{net}}_{100\text{–}500\text{ ms}} + \underbrace{T_{wait}}_{0\text{–}T_e/2} + \underbrace{T_{bft}}_{300\text{–}600\text{ ms}} + \underbrace{T_{confirm}}_{\approx 100\text{ ms}}.$$

Best case: ≈ 1–2 s. High-load case: ≈ 3–5 s.

10.4 Throughput Analysis

Reference validator: RTX 4090, 32-core CPU, 10 Gbps NIC.

- GPU parallel ZK verify: ≈ 500 proofs/s (Groth16, BLS12-381).
- Batch cap: $M_{\max}$ = 300 TX/batch (conservative).
- Epoch duration: T_e = 5 s.
- Throughput: 300/5 = **60 TPS** (single committee).

BFT message complexity grows as $O(n^2)$; threshold signature aggregation keeps finalization bandwidth at $O(n)$ (48-byte shares, one aggregated signature). Recommended committee size: n = 100–300.

10.5 State Growth

- SpentSet: 300 × 17,280 × 365 ≈ 1.9×10^9 entries/year; at 32 bytes/leaf ≈ 60 GB/year full set.
- Light client: SpentSetRoot (32 bytes) + non-membership proof (≈ 1 KB); trivial.

Table 2: System comparison on key axes.

System	Finality	TPS	Privacy	Energy	Ledger
Bitcoin	~60 min	7	Weak	High	Yes
Ethereum	~15 s	15–30	Weak	Low	Yes
Zcash	~75 s	~10	Full	Low	Yes
Algorand	~4.5 s	~1000	Weak	Low	Yes
AXIOM	**2–5 s**	~60*	**Full**	**Minimal**	**No**

* single committee, reference hardware; horizontal scaling possible.

- RSA Accumulator value: constant 256 bytes.
- Pruning: entries older than W epochs ($\approx$ 90 days) are archived; light clients verify against current epoch snapshot.

10.6 Comparison

10.7 Implementation Roadmap

(1) **Phase 1** (3–6 mo): Groth16 circuit (`arkworks-rs`), RSA accumulator, BLS TSig.
(2) **Phase 2** (3 mo): Single-node prototype; SMT SpentSet; Apply() state transition; benchmarks.
(3) **Phase 3** (6 mo): Multi-node BFT; VDF beacon; DKG key rotation; network stress test.
(4) **Phase 4** (3 mo): Tier 0/1 mempool; PoW-lite + stake tickets; dynamic fee market.
(5) **Phase 5** (3 mo): ECVRF sortition; adaptive τ; sybil audit.

Total estimated duration: 18–24 months (small team, production-grade target).

11 SECURITY POLICY AND RESPONSIBLE DEPLOYMENT

This section addresses responsible deployment of Snarc Axiom, independent of the cryptographic security proofs in preceding sections. Cryptographic soundness is a necessary but not sufficient condition for safe deployment of a financial protocol. The policies below bound the impact of any failure—whether cryptographic, implementation, or operational—and define the conditions under which the system escalates, freezes, or migrates.

11.1 Deployment Tiers

Snarc Axiom is defined across three deployment tiers with strictly increasing security requirements and value exposure.

Tier 0 — Researchnet (v1). The protocol described in this paper. Instantiation: Groth16 + RSA-2048 + stake-VRF BFT. Coins carry no monetary value. Purpose: academic analysis, benchmark collection, circuit validation, and adversarial testing. *No real-value assets are issued or redeemable under Tier 0.*

Tier 1 — Testnet (v2). Universal-setup or setup-free proof system (PLONK/Halo2). Hybrid PoW-lite + stake committee admission. Value caps enforced at the protocol level (see Section 11.4). Independent security audit required before any value-bearing token launch. Bug bounty active.

Tier 2 — Mainnet (v3). Setup-free proof system (STARKs or equivalent). Post-quantum signatures (e.g., Dilithium/Falcon) for committee TSig. PQ-resistant accumulator (hash-based state proof or class-group accumulator). Multi-audit and formal verification of critical modules. Full regulatory and legal review per jurisdiction. *Only Tier 2 is suitable for production monetary value.*

11.2 Cryptographic Agility

All cryptographic components are treated as plug-in modules behind stable interfaces:

- **Proof system**: $\Pi.\{\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}\}$ — replaceable without changing the ZK relation $\mathcal{R}$ constraints.
- **Accumulator**: $\mathsf{ACC}.\{\mathsf{Update}, \mathsf{VerifyMem}\}$ — RSA-2048 $\rightarrow$ class-group $\rightarrow$ hash-based, migrated via epoch transition (Section 8).
- **Threshold signature**: BLS12-381 $\rightarrow$ lattice-based TSig, swapped during a scheduled epoch rotation.
- **Hash / PRF**: BLAKE3/Poseidon $\rightarrow$ SHA-3 family or SPHINCS-compatible; parameter-upgradeable.

A migration between tiers is executed as a coordinated epoch transition (Section 8) with a mandatory overlap period of at least W epochs (weak subjectivity window) during which both the old and new proof systems are accepted.

11.3 Threat Escalation and Freeze Policy

Three escalation levels are defined based on observed or credible threat signals.

Level 1 — Yellow (Monitor). Trigger: published theoretical attack, unconfirmed anomaly, or unusual on-chain pattern detected by monitoring. Response: increase audit frequency; convene security committee within 24 h; prepare freeze parameters; no operational change yet.

Level 2 — Orange (Restrict). Trigger: credible proof-of-concept exploit reported, or anomalous state growth/nullifier collision rate exceeding 3σ baseline. Response: (i) halt *MintTx* and *BurnTx* immediately; (ii) reduce per-epoch batch cap $M_{\max}$ to 10% of normal; (iii) notify all validators and auditors; (iv) begin emergency migration preparation.

Level 3 — Red (Full Freeze). Trigger: confirmed exploit, double-finalization observed, or trusted-setup compromise credibly demonstrated. Response: (i) halt all *SpendTx*, *MintTx*, and *BurnTx*; (ii) only *read-only* state queries remain live; (iii) publish incident report within 6 h; (iv) initiate emergency migration or coordinated shutdown.

The freeze condition is enforced by the BFT committee via a threshold-signed *HaltTx*: any t-of-n validator quorum can issue a halt, and no validator will sign new batch proposals until a validated *ResumeTx* (also requiring t-of-n threshold authorization) is produced. *HaltTx and ResumeTx do not confer any ability to reassign coin ownership; they only suspend and resume state transitions under threshold authorization.* Neither transaction can move funds, alter accumulator witnesses, or modify the SpentSet; the Apply function is simply not called until *ResumeTx* is finalized.

11.4 Value Caps

Any Tier 1 deployment must enforce the following, with the enforcement mechanism explicitly specified:

- **Per-transaction cap** ($v \le V_{\max}^{tx}$): enforced *in-circuit* as a range constraint over the committed value field. The ZK proof is unsatisfiable for any $v > V_{\max}^{tx}$; no trusted party can override this.
- **Per-epoch issuance cap** ($\sum_{MintTx \in B_k} v_i \le V_{\max}^{epoch}$): enforced as a *committee validation rule*. Validators maintain a per-epoch issuance counter in the batch proposal; honest validators reject any proposal exceeding the cap. The counter is included in the threshold-signed finalization object C_k and is therefore auditable.
- **Total outstanding supply cap** ($\sum_{cm \in \mathsf{ACC}} v_{cm} \le V_{\max}^{total}$): enforced as a *committee validation rule* backed by an accumulator-derived supply counter in S_k. As with the epoch cap, honest validators reject proposals that would push the counter above $V_{\max}^{total}$.
- **Redemption gate**: BurnTx is disabled by default; enabled only after explicit audit sign-off per deployment. The gate is a committee rule, not a circuit constraint, because enabling it requires no change to the ZK circuit.

11.5 Audit and Bug Bounty Requirements

Tier 0. Internal review only. Open-source release with explicit "experimental, no value" label.

Tier 1. Minimum two independent external security audits of: (i) the ZK circuit and relation $\mathcal{R}$; (ii) the BFT + VDF finality sublayer; (iii) the admission and spam-resistance layer. Bug bounty active from day 1 of public testnet; severity-tiered rewards.

Tier 2. All Tier 1 requirements plus: formal verification of the Apply state transition function (Theorem 6.6) using a proof assistant (e.g., Coq, Lean); hardware wallet and side-channel audit; jurisdiction-specific legal review.

11.6 Quantum Threat Timeline and Migration Trigger

The v1 cryptographic components (Groth16, RSA-2048, BLS12-381) are not post-quantum secure. A sufficiently large fault-tolerant quantum computer would break them via Shor's algorithm. Current consensus among cryptographers places such a machine at 10–20+ years away for cryptographically relevant scale, though this estimate carries significant uncertainty.

Migration triggers are defined as *externally verifiable events*, not subjective forecasts, to ensure objective escalation without reliance on any single party's judgment:

1. **Standards milestone**: NIST formally deprecates RSA-2048 or elliptic-curve cryptography for near-term use (e.g., via a FIPS revision or sunset advisory with a fixed end-of-life date).
2. **Community consensus**: A reproducible break or significant speedup against discrete-log or factoring is documented in a peer-reviewed publication at a major venue (IEEE S&P, CCS, Crypto, Eurocrypt) and independently confirmed by at least one other research group.
3. **Practical break**: A publicly reproducible demonstration breaks a concrete instance of BLS12-381 DL, RSA-2048 factoring, or the Groth16 proof system at any key size used in the deployment.

On any trigger, Level 2 (Orange) is declared within 24 hours, and the migration to a setup-free, PQ-resistant configuration (Tier 2) begins under the epoch rotation protocol (Section 8). Trigger conditions are defined as externally verifiable events (standards milestones or publicly reproducible breaks), not subjective forecasts.

11.7 Responsibility Allocation

This paper describes a cryptographic protocol design. Responsibility allocation for production deployments is as follows:

- **Protocol authors**: specification correctness; honest disclosure of limitations (this section and Section 13).
- **Implementers**: correctness of the implementation against the specification; side-channel hardening; wallet security.
- **Operators**: audit compliance; value-cap enforcement; incident response execution; regulatory compliance.
- **Users**: understanding the risk tier of the deployment they use; not exceeding their own risk tolerance.

No deployment of SNARC AXIOM at any tier carries an implied guarantee of security or suitability for any particular use. The protocol is provided as a research artifact under open-source terms. Use at own risk.

12 RELATED WORK

Privacy-preserving payments. The Zerocash line [1] standardized the nullifier-plus-ZK-proof approach to transaction privacy, and Zcash Sapling [6] industrialized it with production-grade circuit design and a nullifier note commitment scheme. AXIOM's cryptographic core shares this lineage but removes the blockchain substrate entirely. Groth16 [12] and PLONK [8] underpin our "succinct proof, fast verify" requirements.

Compact-state and ledgerless approaches. Utreexo [4] demonstrates that Bitcoin full-node storage can be dramatically compressed by replacing the UTXO set with a dynamic hash accumulator and shifting proof storage to users. AXIOM follows the same "state = compact root" intuition but targets a stronger goal: no total-order log of past transactions is maintained at all. Mimblewimble [13] achieves cut-through compression via Pedersen commitments while retaining a blockchain; AXIOM further eliminates that requirement.

BFT finality and partial synchrony. Our consistency layer is grounded in the partial synchrony model of Dwork–Lynch–Stockmeyer [5]. PBFT [3] establishes the quorum-intersection argument that underpins our Theorems 5.5 and 8.2. HotStuff-class protocols provide the $O(\Delta)$ liveness bound used in Theorem 5.6.

VRF-based sortition. Algorand [9] introduced private VRF sortition for committee selection in a Byzantine agreement context, achieving unpredictability without a central coordinator. AXIOM Section 9 extends this with stake-weighted adaptive sizing, epoch rotation compatibility, and a beacon seed chain that resists grinding (Theorem 9.2). ECVRF is standardized in RFC 9381 [10].

Verifiable Delay Functions. Pietrzak [14] and Wesolowski [15] independently constructed efficient VDFs; Boneh et al. [2] survey both. Axiom uses VDFs in two roles: (i) as an ordering beacon to constrain leader discretion (Section 5), and (ii) as a temporal barrier against long-range attacks (Theorem 8.5).

13 CONCLUSION

This paper presented Axiom, a cryptographic payment protocol that achieves double-spend prevention, full transaction privacy, and fast finality without a global transaction ledger. The core insight is that blockchain's three roles—canonical ordering, nullifier tracking, and coin-validity auditing—can be disaggregated. Coin validity becomes a ZK proof; coin existence becomes an accumulator membership witness; double-spend prevention becomes a nullifier registry; and global consistency becomes a BFT finalization certificate. Together, these replace the ledger with two constant-size state roots and a thin finality layer.

We formalized this design through nine theorems covering correctness, unforgeability, value conservation, double-spend resistance, anonymity, non-malleability, safety, liveness, and atomic state transition. We extended the model to epoch rotation, permissionless VRF sortition, and a composable spam-resistance layer, and provided a concrete instantiation with estimated latency of 2–5 seconds and ≈60 TPS on reference hardware.

Open problems. Four engineering challenges remain for production deployment: (i) stake concentration and plutocracy risk in permissionless admission; (ii) calibration of spam-resistance parameters across heterogeneous deployments; (iii) the trusted-setup versus performance trade-off in accumulator and proof system selection; and (iv) long-term archive and checkpoint operation at scale. We leave formal treatment of these to future work.

REFERENCES

[1] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. 2014. Zerocash: Decentralized Anonymous Payments from Bitcoin. In *2014 IEEE Symposium on Security and Privacy (S&P '14)*. IEEE, 459–474. https://doi.org/10.1109/SP.2014.36

[2] Dan Boneh, Benedikt Bünz, and Ben Fisch. 2018. A Survey of Two Verifiable Delay Functions. Cryptology ePrint Archive, Report 2018/712. https://eprint.iacr.org/2018/712

[3] Miguel Castro and Barbara Liskov. 1999. Practical Byzantine Fault Tolerance. In *Proceedings of the 3rd USENIX Symposium on Operating Systems Design and Implementation (OSDI '99)*. USENIX Association, New Orleans, LA, 173–186. https://pmg.csail.mit.edu/papers/osdi99.pdf

[4] Tadge Dryja. 2019. Utreexo: A Dynamic Hash-Based Accumulator Optimized for the Bitcoin UTXO Set. Cryptology ePrint Archive, Report 2019/611. https://eprint.iacr.org/2019/611

[5] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. 1988. Consensus in the Presence of Partial Synchrony. *Journal of the ACM* 35, 2 (April 1988), 288–323. https://doi.org/10.1145/42282.42283

[6] Electric Coin Company and Zcash Foundation. 2022. Zcash Protocol Specification, Version 2022.3.8 (Sapling). Technical Specification. https://zips.z.cash/protocol/protocol.pdf

[7] Paul Feldman. 1987. A Practical Scheme for Non-interactive Verifiable Secret Sharing. In *28th Annual Symposium on Foundations of Computer Science (FOCS '87)*. IEEE, 427–438. https://doi.org/10.1109/SFCS.1987.4

[8] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. 2019. PLONK: Permutations over Lagrange-Bases for Oecumenical Noninteractive Arguments of Knowledge. Cryptology ePrint Archive, Report 2019/953. https://eprint.iacr.org/2019/953

[9] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. 2017. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '17)*. ACM, 51–68. https://doi.org/10.1145/3132747.3132757

[10] Sharon Goldberg, Leonid Reyzin, Dimitrios Papadopoulos, and Jan Vcelak. 2023. Verifiable Random Functions (VRFs). Request for Comments 9381, Internet Engineering Task Force (IETF). https://doi.org/10.17487/RFC9381

[11] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. 2021. Poseidon: A New Hash Function for Zero-Knowledge Proof Systems. In *30th USENIX Security Symposium*. USENIX Association, 519–535. https://eprint.iacr.org/2019/458

[12] Jens Groth. 2016. On the Size of Pairing-Based Non-interactive Arguments. In *Advances in Cryptology – EUROCRYPT 2016 (Lecture Notes in Computer Science, Vol. 9666)*. Springer, 305–326. https://doi.org/10.1007/978-3-662-49896-5_11

[13] Tom Elvis Jedusor. 2016. Mimblewimble. Unpublished whitepaper, distributed via the Bitcoin Research IRC channel. https://download.wpsoftware.net/bitcoin/wizardry/mimblewimble.txt Pseudonymous authorship; identity unverified.

[14] Krzysztof Pietrzak. 2019. Simple Verifiable Delay Functions. In *10th Innovations in Theoretical Computer Science Conference (ITCS 2019) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 124)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 60:1–60:15. https://doi.org/10.4230/LIPIcs.ITCS.2019.60

[15] Benjamin Wesolowski. 2019. Efficient Verifiable Delay Functions. In *Advances in Cryptology – EUROCRYPT 2019 (Lecture Notes in Computer Science, Vol. 11478)*. Springer, 379–407. https://doi.org/10.1007/978-3-030-17659-4_13