The Horizon Protocol

A Stateless Post-Quantum Blockchain based on the Holographic Principle

Aaron M. Schutza

February 19, 2026

Abstract

We introduce the **Horizon Protocol**, a stateless blockchain architecture designed to solve the scalability issues inherent in post-quantum cryptography. While lattice-based signatures (such as our proposed Jordan-Dilithium) offer security against quantum adversaries, their large size ($\approx$ 2KB) creates a storage bottleneck in traditional UTxO models. Horizon utilizes a **Holographic State Model**, where the consensus layer stores only a 32-byte state root (the "Horizon"), while the bulk data is maintained by users via cryptographic witnesses. The integrity of the state is secured by **GSH-256**, a novel hash function based on the non-associative sponge construction over Sedenions ($\mathbb{S}_{16}$), providing Topological Impedance against quantum collision attacks.

1 Introduction

1.1 The Post-Quantum Scalability Problem

The transition to post-quantum cryptography (PQC) presents a trilemma. Algorithms secure against Shor's algorithm, such as Lattice-based or Hash-based signatures, are significantly larger than their classical counterparts (ECDSA).

- **Bitcoin (ECDSA):** Signature $\approx$ 64 bytes.
- **Jordan-Dilithium (APH):** Signature $\approx$ 2500 bytes.

In a standard stateful blockchain, verifying nodes must store the entire Unspent Transaction Output (UTxO) set. Increasing the signature size by 40$\times$ exacerbates bandwidth and storage requirements to unsustainable levels.

1.2 The Holographic Solution

The Horizon Protocol decouples *validation* from *storage*. We treat the blockchain state according to the Holographic Principle: the information of the "Bulk" (the UTxO set) is encoded on a lower-dimensional boundary, the "Horizon" (the State Root).

- **Validators:** Store only the Horizon (32 bytes).
- **Users:** Store the Witness (Merkle Branch) for their own assets.
- **Signatures:** Exist only transiently in the transaction data to authorize state transitions; they are never stored in the Horizon.

2 Cryptographic Primitives

2.1 GSH-256: Geometric Stiffness Hash

Standard Merkle trees rely on SHA-256 or Poseidon. We introduce **GSH-256**, a hash function defined by a Sedenion Sponge construction.

- **Domain:** Sedenions $\mathbb{S} \cong \mathbb{O} \times \mathbb{O}$ (Dimension 16).
- **Non-Associativity:** The core compression function relies on the Sedenion Associator $[X, Y, Z] = (XY)Z - X(YZ)$.
- **Hardness:** Finding collisions requires solving the *Associator Inverse Problem*, which is computationally irreducible due to the chaotic trajectory of non-associative operations (Topological Impedance).

2.2 Jordan-Dilithium Signatures

Transactions are authorized using the Jordan-Dilithium scheme over the Albert Algebra $J_3(\mathbb{O})$. This scheme uses a scalar challenge to bypass Artin's Theorem during verification, ensuring validity without compromising the non-associative hardness of the secret key.

3 The Horizon Architecture

3.1 The Accumulator (Sparse Merkle Tree)

The Global State is a Sparse Merkle Tree (SMT) of depth $H = 64$, effectively covering an address space of 2^{64}.

$$Root_{Level} = GSH(Child_L \parallel Child_R) \tag{1}$$

The leaf nodes contain the hash of the UTxO data:

$$Leaf_i = GSH(TxID \parallel Owner_{PK} \parallel Amount) \tag{2}$$

3.2 The Transaction Structure

A transaction Tx in Horizon does not reference a stored UTxO. Instead, it provides a proof of existence.

$$Tx = \{U, \pi, \sigma, U_{new}\} \tag{3}$$

Where:

- U: The UTxO being spent.
- π: The Witness (Merkle Sibling Path) proving $U \in Horizon_{Current}$.
- σ: The Jordan-Dilithium signature verifying ownership of U.
- U_{new}: The new UTxO to be created (optional, for output).

4 Stateless Validation Logic

A validator node holds only the current root R_{curr}. Upon receiving Tx, it performs the following O(1) storage operations:

5 Performance Analysis

5.1 Storage Efficiency

While the transaction size increases due to the inclusion of the Witness (π), this is a bandwidth cost, not a permanent storage cost. The blockchain history (the chain of signatures) can be pruned, as only the current Horizon Root is required for consensus.

Algorithm 1 Horizon State Transition

Input: R_{curr}, Tx
Step 1: Verify Signature
  Check $Verify(U.Owner, H(U), Tx.\sigma)$
if Invalid **then return Reject**
end if
Step 2: Verify Witness
  Compute $R_{calc} = MerkleRoot(U, Tx.\pi)$
if $R_{calc} \neq R_{curr}$ **then return Reject (UTxO not in state)**
end if
Step 3: Update State (The Burn)
  Compute $R_{next} = MerkleRoot(EmptyHash, Tx.\pi)$
Step 4: Update State (The Mint)
  (Logic to add U_{new} to empty leaf k)
return R_{final}

Metric	Standard UTxO	Horizon (Stateless)
Validator Storage	$\approx$ 50 GB (growing)	**32 Bytes (Fixed)**
Tx Size (PQ)	$\approx$ 2.5 KB	$\approx$ 4.5 KB (Sig + Witness)
Bandwidth	High	Moderate
Bootstrap Time	Days	**Seconds**

Table 1: Comparison of Architectures

6 Conclusion

The Horizon Protocol resolves the tension between Post-Quantum security and blockchain scalability. By adopting a Holographic State Model secured by Geometric Stiffness Hashing, we enable a robust, decentralized financial layer that remains secure against quantum adversaries without succumbing to state bloat.