BETON & BSH

Whitepaper

On-Chain BTC Price Wagering and the BSH Token Economy on Solana

Version	1.0
Published	April 2026
Network	Solana
Status	Public Release

Disclaimer

This whitepaper is published for informational purposes only. Nothing contained herein constitutes financial advice, an offer to sell or purchase securities, or a solicitation to invest in any digital asset. The protocol mechanics, tokenomics, and design descriptions reflect the state of the software at the time of writing and may be updated as the project evolves.

Participating in any decentralized protocol involves risk, including smart contract vulnerabilities, oracle dependency, market volatility, and regulatory uncertainty. Readers are encouraged to conduct their own research and consult qualified advisors before interacting with any protocol described in this document. Past performance of any market or asset is not indicative of future results.

Introduction

We are the team behind BETON and BSH. We built this project because we believe two things strongly.

First: price-prediction markets are one of the clearest expressions of on-chain value — peer-to-peer, transparent, and inherently suited to a permissionless blockchain. But almost every implementation we've seen falls short in the same ways. Funds sit in shared custodial pools rather than isolated per-bet escrow. Oracle prices are accepted without verification checks. Outcomes depend on off-chain servers that users have no way to independently verify.

Second: most project tokens have no durable reason to hold value beyond speculation. They exist separately from the protocol that issued them, disconnected from throughput, usage, or any tangible backing.

We designed BETON and BSH to solve both problems at once — and to do it with code that is fully on-chain, publicly verifiable, and auditable by anyone.

This whitepaper explains what we built, how it works, and why we made the design choices we did.

Table of Contents

1. The Problem
2. Our Solution at a Glance
3. BETON — The On-Chain Wagering Protocol
4. BSH — The Project Token
5. The Economic Flywheel
6. Fee Structure and Distribution
7. The Jackpot System
8. Security Guarantees
9. A User's Journey
10. Technical Architecture Overview
11. Token Economics
12. Conclusion

---

1. The Problem

Wagering platforms hold your funds

Most price-prediction and wagering platforms operate with a centralized or semi-centralized custody model. Your SOL or tokens sit in a shared contract account controlled by the platform operator. If the contract has a bug, is upgraded maliciously, or is abandoned, users have limited recourse. You trust the platform; you do not trust the code.

Oracle prices are easy to manipulate — if you let them be

Settling a bet based on a price requires a reliable, tamper-resistant price source. Most platforms either use an easily manipulable on-chain Pyth account without validation, or rely on an off-chain server to report the closing price. Both approaches leave the door open to stale, replayed, or artificially injected prices. These are not theoretical risks — oracle manipulation is one of the most common attack vectors in DeFi.

Project tokens rarely reflect protocol value

The crypto space is full of protocols with utility tokens that have no structural connection to the protocol they represent. The token price is driven by sentiment, not by any on-chain mechanism. When usage grows, the token does not necessarily benefit. When usage falls, there is nothing to support it. The disconnect is a feature of bad design, not an inevitability.

---

2. Our Solution at a Glance

BETON is a non-custodial BTC price-direction wagering protocol. Every bet's funds live in a dedicated escrow account controlled exclusively by the BETON program. Settlement uses the Pyth pull-oracle with strict on-chain validation: staleness windows, confidence-interval checks, feed ID verification, and replay prevention. No off-chain server can influence the outcome. No admin key can touch the funds.

BSH is our project token. Its backing is not a promise — it is an on-chain mechanism. Every time a BETON bet is settled or refunded, the protocol automatically routes a fee slice directly into the BSH treasury vault. The transfer is enforced by the settlement instruction itself, and the instruction fails if the transfer does not happen. BSH backing grows as a direct, verifiable function of BETON activity.

The two programs share no runtime state and no execution path. Their connection is economic: a validated SOL transfer from BETON into the BSH vault on every settlement. This keeps both programs independently auditable while creating a compounding relationship between wagering activity and token value.

---

3. BETON — The On-Chain Wagering Protocol

What BETON does

BETON lets any two participants wager on whether the BTC/USD price will be higher or lower at a future point than it is right now. The outcome is determined entirely by Pyth oracle data. There is no house, no counterparty, and no admin who can influence the result. The smart contract holds the stakes, validates the price, and distributes the winnings — without any human intervention.

How a bet works

Step 1 — Creating a bet. The first participant (the creator) deposits their stake, chooses a duration, and picks a direction: Bullish (price goes up) or Bearish (price goes down). At this moment, BETON reads the current BTC/USD price from Pyth and locks it into the bet as the opening reference price. This price is verified against Pyth's confidence and staleness requirements before being accepted.

Step 2 — Accepting a bet. Any other participant can accept the open bet by depositing an equal stake. They automatically take the opposite direction. Once a bet is accepted, the settlement clock starts.

Step 3 — Settlement. After a mandatory delay, a settlement window opens. During this window, either participant can submit the settlement transaction. BETON fetches a fresh Pyth BTC/USD price, compares it to the locked opening price, and pays the winner. Settlement is a single atomic on-chain operation — it cannot be partially applied or interrupted.

Step 4 — Cleanup. After settlement, the bet account remains on-chain for 15 days (allowing time for review or downstream integrations) before it can be closed and the rent reclaimed.

Duration options

Duration	Accept window	Delay before settlement	Settlement window
Quick	20 minutes	20 minutes	20 minutes
Standard	1 hour	1 hour	1 hour
Long	3 hours	3 hours	3 hours

The delay window is mandatory and cannot be bypassed. This means BETON is a forward prediction market — participants are predicting where BTC will be in the future, not reacting to the current tick.

Outcome rules

- If BTC closes **above** the opening price: the **Bullish** side wins.
- If BTC closes **below** the opening price: the **Bearish** side wins.
- If BTC closes at **exactly** the opening price: the participant who calls settlement wins the pot and collects the bonus vault balance. No draw, no refund — the protocol rewards the person who acts.

What happens if a bet expires

If an accepted bet is not settled within the settlement window, either participant can trigger a refund. The same 2% fee is applied, and both sides receive 98% of their stake back. The fee paths still receive their portion — this discourages participants from deliberately ignoring bets to avoid fees.

If a bet was created but never accepted, the creator can cancel it after the accept window expires and receive a full refund with no fee.

Rate limiting

To protect against spam and jackpot farming, each user is limited to four bet actions per hour. This limit is enforced on-chain — no client-side bypass is possible.

---

4. BSH — The Project Token

What BSH is

BSH is the native token of the BETON protocol. It is a fixed-supply Solana SPL token with a SOL-backed treasury vault. Every BSH token represents a proportional claim on the SOL held in the vault. You can buy BSH by depositing SOL into the vault, and sell BSH to receive SOL back from the vault. The price in both directions is determined by the ratio of vault SOL to total supply.

Fixed supply, forever

BSH was minted once at launch: exactly **100,000 tokens**, with zero decimal places. After minting, both the mint authority and freeze authority were permanently revoked on-chain. No one — including us — can ever create additional BSH. The supply is mathematically fixed.

No admin control over the vault

The BSH SOL vault is a program-owned account. No wallet key, including ours, can withdraw from it directly. The only operations the program supports are buying BSH (SOL flows in) and selling BSH (SOL flows out), each with mandatory slippage protection. The vault cannot be drained below a minimum floor in a single transaction, and neither the buy nor the sell can exhaust the vault's BSH or SOL inventory completely.

How to acquire BSH

Buy (SOL → BSH): Deposit SOL into the vault and receive BSH at the current share price. The price you pay is calculated using the vault's SOL balance *after* your deposit is received — this detail prevents atomic buy-sell sandwich attacks.

Sell (BSH → SOL): Return BSH to the vault and receive the proportional SOL backing. The amount you receive is calculated from current vault SOL divided proportionally by total supply.

Both directions require you to specify a minimum acceptable output. If the price moves unfavorably between when you build the transaction and when it executes, the transaction fails rather than executing at a worse rate.

Immutable metadata

BSH token metadata — name, symbol, and URI — was created on-chain through Metaplex before the mint authority was revoked. Since the mint authority no longer exists, the metadata cannot be changed. What you see is what it will always be.

---

5. The Economic Flywheel

The most important design property of BETON and BSH together is that they form a self-reinforcing loop enforced entirely by on-chain code.

Every time a BETON bet is settled or refunded, **0.5% of the total pot** is transferred directly into the BSH SOL vault. This is not a discretionary donation from our treasury. It is a mandatory fee transfer that the settlement instruction executes and verifies. If the transfer fails or the wrong amount is moved, the entire settlement instruction reverts.

The effect is cumulative:

1. More bets are placed and settled on BETON.
2. More SOL flows into the BSH vault.
3. The vault's SOL backing per token increases.
4. BSH buyers receive more SOL per token when they sell.
5. Deeper backing attracts more BSH holders.
6. BSH holders have an incentive to participate in and promote BETON.

This flywheel is not a narrative or a roadmap promise. It is a property of the code, active from the moment the protocol goes live.

```
BETON activity → SOL into BSH vault → deeper token backing → stronger
BSH → more BETON participation
```

6. Fee Structure and Distribution

BETON applies a **2% total fee** on every settled or refunded bet. No fee is charged on cancellations (unaccepted bets).

For a bet where both sides stake amount **A**:

Quantity	Value
Total pot	2A
Total fee (2%)	0.04A
Winner payout	1.96A

The 2% fee is split equally into four 0.5% slices:

Recipient	Purpose
Jackpot pool	Funds the periodic jackpot payout
BSH SOL vault	Increases BSH token backing
Protocol treasury (1 of 2)	Ongoing development and operations
Protocol treasury (2 of 2)	Reserve

Every fee transfer is verified on-chain after it executes. If the actual lamport movement does not match the computed slice, the instruction fails. This means the fee distribution is not a best-effort operation — it is an invariant that the program enforces.

---

7. The Jackpot System

0.5% of every pot goes into the jackpot pool. The jackpot is not a random lottery — it rewards consistent winning.

How it works

Winning participants accrue win counts stored in their on-chain activity record. When a user's cumulative win count reaches the current threshold, they trigger a jackpot payout and receive the full jackpot pool balance in addition to their normal winnings.

After each jackpot payout, the threshold doubles. This means early jackpots are relatively easy to reach, and each subsequent one requires progressively more wins. When the threshold grows beyond the configured maximum, it resets to 1 and a new jackpot cycle begins — without requiring any rewrite of existing user data.

Tie-settlement bonus

When a bet ends in a tie (opening price equals closing price), the jackpot tracking is skipped. Instead, the participant who calls settlement receives the entire bonus vault balance — in addition to the normal post-fee pot. The bonus vault is continuously refilled by rent reclaimed when bet accounts are closed. This makes settlement during a tie a directly rewarded action rather than a courtesy.

---

8. Security Guarantees

Your funds are not in our custody

Every bet's stakes are held in a dedicated on-chain escrow account controlled by the BETON program. Even we cannot withdraw those funds. The only operations that can move them are the program instructions: settlement pays the winner, refund returns stakes symmetrically, and cancel returns the creator's stake in full. There is no shared pool and no admin withdrawal function.

Oracle prices are validated on-chain

Before BETON accepts any price from Pyth, it checks:

- The price account is owned by the Pyth receiver program (not just any account)
- The feed ID matches the configured BTC/USD feed exactly
- The price is non-zero and was published recently (within a strict freshness window)
- The confidence interval is within an acceptable range relative to the price
- The attestation meets minimum verification requirements

These checks happen inside the BETON program, not in the client application. Even if someone submits a transaction with a forged or stale price account, the program rejects it.

A settled bet cannot be settled again

When a bet is settled, the program stores a cryptographic fingerprint of the prices and timestamp used. Any subsequent attempt to settle the same bet — even with a valid Pyth account — fails because the fingerprint field is already filled. No global registry or shared nonce required.

Concurrent manipulation is prevented

Each bet carries a lock that is set at the beginning of any mutating operation and cleared regardless of whether the operation succeeds or fails. This prevents two transactions from simultaneously modifying the same bet and ensures locks are never left permanently set even in error conditions.

The BSH vault cannot be arbitrage-drained

The BSH buy instruction calculates the output amount using the vault balance *after* the SOL deposit is received. This means a user who buys and immediately sells in the same transaction gets back exactly what they put in — no risk-free profit is possible. Neither swap can reduce the vault's token inventory or SOL balance to zero; minimum-floor constraints are enforced on both sides.

9. A User's Journey

Here is what a typical interaction with BETON looks like from start to finish.

Connect your Solana wallet. The application supports standard Solana wallet adapters. Your keys never leave your device.

Register your activity account. The first time you interact with BETON, you create a small on-chain account that tracks your bet history and jackpot progress. This is a one-time setup.

Create or accept a bet. Choose a duration, stake the amount you want, and pick your direction. Or browse open bets and accept one that interests you, matching the existing stake exactly.

Wait for the settlement window. After a mandatory delay, the settlement window opens. Either participant can call settlement. The application automatically fetches a fresh BTC/USD price from Pyth and submits the settlement transaction.

Collect your winnings. If you won, the smart contract pays your winnings directly to your wallet in the same settlement transaction. No claim step, no withdrawal delay beyond the transaction confirmation.

Buy or sell BSH. Visit the BSH swap interface, deposit SOL to receive BSH at the current vault price, or sell BSH to receive SOL. Both directions execute atomically on-chain with slippage protection.

---

10. Technical Architecture Overview

This section provides a concise technical summary for readers who want a deeper understanding of how the system is built. A full developer reference is published separately.

System architecture

```
flowchart LR
    App[Web Application] --> SDK[Typed On-Chain SDK]
    SDK --> Beton[BETON Program]
    SDK --> BSH[BSH Program]

    Beton --> Escrow[Per-bet Escrow Account]
    Beton --> Jackpot[Jackpot Pool]
    Beton --> BonusVault[Bonus Vault]
    Beton --> Treasury[Protocol Treasury]
    Beton --> BshVault[BSH SOL Vault]

    BSH --> BshVault
    BSH --> TokenVault[BSH Token Inventory]

    Pyth[Pyth Price Oracle] --> App
    App --> PythReceiver[Pyth Receiver Program]
    PythReceiver --> Beton
```

Two independent programs

BETON and BSH are two separate Solana programs. They share no code, no runtime state, and no common execution path. Their integration is limited to a single verified SOL transfer from BETON into the BSH vault during settlement.

This design means either program can be audited independently. A vulnerability in one does not automatically compromise the other.

Per-bet escrow

Each bet gets its own dedicated SOL escrow account, derived from the bet's unique on-chain address. There is no shared deposit pool. Multiple bets settling at the same time cannot interfere with each other's funds.

Pyth pull-oracle integration

BTC/USD prices are sourced from Pyth Network's pull-oracle system. The application fetches signed price updates from Pyth's Hermes service and posts them to the Pyth receiver program on-chain. BETON then reads from the receiver's on-chain price account — and validates it thoroughly before accepting any value from it.

Because the price data payload is large, settlement requires two sequential transactions: one to post the Pyth price update, and one to settle the bet and reclaim the temporary account's rent. The application handles this transparently.

BSH pricing model

BSH does not use a constant-product AMM formula. Instead, each token represents a proportional share of the vault's SOL balance:

- **Buying:** BSH received = SOL deposited × total supply ÷ vault after deposit
- **Selling:** SOL received = BSH sold × vault SOL ÷ total supply

This formula behaves like a tokenized treasury share. As the vault accumulates more SOL from BETON fee inflows, each BSH token's proportional SOL claim grows.

On-chain verification

Both programs include embedded test suites executed against an in-process Solana VM. BETON tests cover the full bet lifecycle including edge cases: rate limiting, oracle rejection, replay prevention, tie resolution, jackpot payout, and account cleanup. BSH tests cover supply integrity, pricing math, and vault-preservation invariants. The application includes mutation-level tests for the key wagering flows.

11. Token Economics

Property	Value
Total supply	100,000 BSH
Decimals	0 (whole tokens only)
Mint authority	Permanently revoked
Freeze authority	Permanently revoked

Initial distribution

The tokens will be distributed as follows:

- **Public Sale** - 80,000 BSH
- **Private Sale** - 10,000 BSH
- **Team** - 6,000 BSH
- **Bounties** - 4,000 BSH

Vault backing

The BSH SOL vault starts seeded by the treasury and grows through protocol activity. Every BETON settlement and refund contributes 0.5% of the pot to the vault. This is a continuous, programmatic source of backing that scales directly with usage.

Token utility

- **Store of value backed by protocol activity.** Accumulated protocol fee inflows.
- **Liquidity participation.** Holding BSH means participating in the treasury-share model.
- **Community access.** BSH holders are the core community of the BETON ecosystem.

---

12. Conclusion

BETON is the protocol we would have wanted to use ourselves: non-custodial, oracle-validated, with no platform key that can touch user funds and no off-chain server that can change the outcome. Every guarantee we describe in this document is enforced by code that is deployed on Solana.

BSH is the token we believe a protocol-backed asset should be: fixed supply, irreversible initialization, and a vault whose SOL balance grows as a direct, verified function of protocol usage — not from promises or manual management.

The two programs are designed to strengthen each other over time. More wagering means more backing. More backing means a stronger token. A stronger token means more participants with a vested interest in the protocol's success.