Vibechain: A Shared Underwriting Protocol for Infinite World Assets

Beb, Inc.
Palo Alto, California
Draft
June 17, 2026

Abstract. Vibechain is a protocol for creating asset economies backed by one shared stablecoin underwriting pool. An asset is defined by an immutable manifest: a controller, fixed entry price, weighted value table, and metadata commitment. Users buy entries against the manifest hash. The contract reserves pool capacity, requests entropy from an immutable adapter, and lets any caller finalize settlement. A nonzero settlement mints a transferable ERC-721 claim redeemable for stablecoin until expiry. The protocol's central rule is a solvency invariant: contract balance must always cover unresolved reservations, live claim values, reserved settlement tips, pending refundable prices, and controller collateral.

1. Introduction

Blockchains made asset creation and ownership permissionless, but they did not make asset liquidity automatic. A developer can define a token, item, or game object; the hard part is still funding a liquid exit and proving that settlement follows a fixed rule rather than a server promise.

Vibechain separates the asset description from the money. The developer commits to the rules in a manifest. The protocol holds the funds, prices the maximum loss exposure, runs settlement, and issues transferable claims. The developer cannot withdraw pool funds, change a live manifest, choose outcomes, or redirect settled value.

The core surface is deliberately small: instantiate a manifest, add or withdraw liquidity, buy entries, finalize settlement, redeem or expire claims, and close a table. Application inventory, metadata, delivery, and interface logic live above the core.

2. Manifests and Claims

The asset identifier is the hash of ABI-encoded manifest bytes. The manifest commits to a controller, a fixed entry price `P`, a list of weighted bands, and metadata. Each band has probability weight `ppm_i` and claim value `u_i`. Weights must sum to one million parts per million. One protocol unit is one stablecoin, so a band value is directly a stablecoin claim value.

The table statistics used by the contract are

$$r_B = \mathrm{ceil}(10000 \cdot \Sigma_i(ppm_i \cdot u_i) / (1000000 \cdot P))$$
$$r = r_B / 10000$$
$$M = \max_i u_i$$

where `r_B` is the return ratio in basis points and `M` is the maximum claim value. The public shared-pool wrapper also rejects tables with `M > 50P`; this is a deployment policy, not a change to the core accounting.

A nonzero settlement mints one ERC-721 claim to the entry holder. The claim stores a stablecoin value and an expiry. It may be transferred, redeemed by its owner, redeemed by an approved operator, or expired after its deadline. Redemption burns the claim and pays its value from the pool to the redeemer.

3. Pool Accounting

The pool separates general liquidity from controller collateral. General liquidity is the shared underwriting capital. Controller collateral belongs to a controller and backs that controller's live tables under the margin rule.

Let

$$O = \text{totalReserved} + \text{totalWinUnits} + \text{totalReservedTips} + \text{totalPendingPrice}$$
$$F = \max(\text{stableCoin}_{\text{balance}} - O - \text{totalAssetCollateral}, 0)$$

where `F` is `freePot()`. The post-call solvency invariant is

$$\text{stableCoin}_{\text{balance}} \geq O + \text{totalAssetCollateral}$$

`totalReserved` covers unresolved entries. `totalWinUnits` covers live claims. `totalReservedTips` covers settlement tips. `totalPendingPrice` covers prices that must be refundable until settlement or timeout. `totalAssetCollateral` keeps controller collateral outside general pool net asset value.

General liquidity withdrawals are bounded by free liquidity. Controller collateral withdrawals are bounded by the margin rule and by the backing lock defined below. Claim redemption, refunds, settlement tips, and bounded liquidity or collateral withdrawals are the only stablecoin outflows.

4. Margin and Spread

For a table with maximum claim value `M`, return ratio `r_B`, and live margin slope `gamma_B`, the controller collateral requirement is

$$C = 0, \qquad \text{if } r_B + \text{gamma}_B \leq 10000$$
$$C = \text{ceil}(M \cdot (\text{gamma}_B - (10000 - r_B)) / \text{gamma}_B), \quad \text{otherwise}$$

This is the integer form of

$$C(M,r) = M \cdot (1 - (1 - r) / \text{gamma})^{+}$$

where `x^+ = max(x,0)`. At buy time, each controller must satisfy

$$\text{sum } C(\text{live tables}) \leq \text{posted collateral}$$

On collateral withdrawal, unresolved backing is included:

$$\text{sum } C(\text{live tables}) + \text{sum locked}(\text{live tables}) \leq \text{posted collateral}$$

The gross spread for one entry is computed from the rounded return ratio:

$$e = \text{floor}(P \cdot (10000 - r_B) / 10000)$$
$$d = \text{floor}(e \cdot C / M)$$
$$s = e - d$$
$$b = P - d - s$$

Here `d` is credited to the controller on settlement, `s` is the pool share, and `b` is expected-value backing. The settlement tip is

$$t = \text{floor}(P \cdot \text{settleTipBps} / 10000), \qquad \text{require } s \geq t$$

Before integer rounding, the split is proportional to risk borne:

$$d / e = C / M$$
$$s / e = 1 - C / M$$

Thus a controller that posts the maximum-loss capital receives the corresponding spread, while a controller that shifts exposure to the pool gives that spread to the pool.

5. Reservation, Settlement and Solvency

The full entry price is escrowed as `totalPendingPrice` until the entry settles or refunds. For each entry in a batched buy, the contract reserves against the current free pool after reserving the settlement tip:

$$F'_k = \max(F_k - t, 0)$$
$$cap_k = \text{floor}(kappa_B \cdot F'_k / 10000)$$
$$R_k = \min(M, cap_k)$$
$$F_{k+1} = \max(F'_k - R_k, 0)$$

The buy reverts if `cap_k < 1` or if `R_k` is below the caller's `minReservedPerRoll`. This sequential reservation is the contract-exact form of the pool cap. It prevents a batch from pricing every entry against the same free liquidity.

When entropy is fulfilled, settlement derives

$$seed = H(providerRandom, blockSalt, rollId, holder, assetId)$$
$$drawKey = H(assetId, rollId, holder)$$
$$x = H(drawKey, seed) \bmod 1000000$$

The selected band is the first cumulative `ppm` interval containing `x`. If the selected band has value `u_i`, the claim value is

$$u = \min(u_i, R_k)$$

If `u` is nonzero, the contract mints a claim for `u` stablecoin and moves `u` from open reservation to live claim backing. The unused reservation is released. The controller share is credited to controller collateral, the settlement tip is paid to the finalizing caller, and the pending price is recognized.

If entropy does not arrive before the timeout, the holder is refunded and the entry's reservation, tip, pending price, and backing lock are released. If optional blockhash hardening is enabled and the target blockhash expires before settlement, the same refund path is used.

6. Backing Lock

Expected-value backing cannot be withdrawable while entries are unresolved. On buy, each entry increases the table's backing lock by `b`. On settlement, the lock releases by `b`. On timeout refund, the buy-side backing increase is removed. A controller withdrawing collateral must satisfy the margin rule plus the current unresolved backing lock.

This closes the ledger gap where a controller could otherwise collect interim backing, withdraw it, and leave the pool with the tail event. The lock is not an extra user-facing state machine; it is a withdrawal bound over existing pending entries.

7. Entropy and Verifiable Randomness

The core contract binds one entropy adapter at deployment. Buying an entry calls the adapter for a provider random word. Only the adapter can fulfill the entry's entropy callback. The application developer contributes no secret and has no settlement-time choice.

By default `blockSalt = 0`. If the post-entropy delay parameter is nonzero, the callback snapshots a future block and settlement mixes that blockhash into the seed after it becomes available. This hardens settlement after the provider word is public, but the primary random word is still the adapter's responsibility.

Every finalized entry stores a transcript containing the seed, claim id, units, settled flag, and refund flag. Applications can read this storage instead of trusting an event relay.

8. Lifecycle

Closing a table is one-way. It stops new buys immediately. Pending entries still settle or refund under the same rules. Once the pending count reaches zero, the table is removed from the controller's live exposure set, freeing its collateral requirement and live-table slot. The same manifest id cannot be instantiated again; a relaunch uses a different manifest hash.

The current public wrapper adds two liquidity policies around the core: general LP deposits are blocked while unresolved risk exists, and general LP withdrawals must be requested before execution. These rules protect share accounting and do not alter the settlement or margin formulas above.

9. Security Considerations

The safety properties are mechanical: manifests are immutable by hash, controllers must consent before tables enter their exposure set, settlement and expiration are permissionless, claim redemption uses ERC-721 ownership and approval, entropy fulfillment is adapter-gated, withdrawals are bounded, and the solvency invariant is checked after every state-changing verb.

The main residual assumptions are entropy-provider correctness, chain liveness, and sufficient free liquidity for the table sizes users accept. Provider failure refunds pending entries, but provider bias is outside the core contract except for optional blockhash hardening and public statistical auditability of settlements against committed weights.

10. Conclusion

Vibechain is an underwriting protocol for software-defined asset economies. A manifest commits the table. A shared stablecoin pool provides bounded capacity. A collateral curve prices maximum-loss exposure. Adapter entropy and permissionless finalization settle entries. Nonzero settlements become transferable claims with a stablecoin exit.

The result is a neutral money layer with one central invariant: every live reservation, claim, tip, pending refundable price, and controller collateral balance is covered by contract-held stablecoin.