Overcharge

Yield without a token-bound account: an NFT that earns Robinhood Chain trading fees without leaving Solana, and the attested link that makes the claim checkable.

Degen Dev — The Black Bull

Draft v0.1 (last updated 2026-08-09)

Abstract

Overcharge is a cross-chain activation system pairing a Solana NFT collection, The Black Bull, with a Robinhood Chain token, $BULL, whose trading fees fund holder rewards. Its central design decision is that the NFT never moves: no wrapped representation, no bridge escrow, no synthetic mirror on Robinhood Chain that could desync from the real asset. Instead, a holder submits two signatures, one from the Solana wallet holding the NFT and one from the Robinhood Chain wallet that will lock $BULL, and an attestor folds verified links into a Merkle root posted on-chain, so any specific link can be checked against public state without trusting a database. Locking $BULL against a proven link activates the NFT into one of five reward tiers; half the locked amount burns immediately, and unlocking is never gated. Reward eligibility is fixed at the start of each distribution epoch by a balance snapshot, closing the obvious way to game a payout by locking just before it and withdrawing just after. This document describes the link attestation scheme, the activation and snapshot mechanics, the distribution formula, the threat model, and the staged path from a single attestor to a threshold-signed and eventually natively-verified link layer.

1. Introduction

Every system that rewards an NFT for activity happening on a chain the NFT wasn't issued on needs an answer to one question: how does the payer know which wallet, on that other chain, actually belongs to the NFT's holder.

On a single EVM chain the answer already exists. ERC-6551 gives an NFT its own address, so the NFT simply *is* the wallet, and paying it is paying an address like any other. Projects built this way, StonkBrokers among them, activate an NFT by locking the project token directly into its token-bound account and pay rewards into that same address. No linking step is required because there is nothing to link.

The Black Bull does not have that option. It is a Metaplex Core asset on Solana, a chain with no token-bound account standard and no way for an NFT to hold an EVM balance. Two designs remain. Bridge the NFT itself: wrap it, escrow the original on Solana, mint a synthetic

representation on Robinhood Chain, and activate the synthetic. This concentrates custody risk in a bridge escrow exactly where the native design has none, and creates a wrapped state that can desync from who actually holds the real asset. Or bridge nothing, and prove the relationship instead. Overcharge takes the second path.

2. Design principles

1. **The NFT never leaves Solana.** No wrapping, no escrow, no synthetic mirror that can desync from the real asset.
2. **A link is a claim, not a transfer.** Proving a Solana wallet and a Robinhood Chain wallet belong to the same holder moves no asset and grants no custody in either direction.
3. **Rewards come from realized fees, never emissions.** Overcharge does not mint $BULL to pay yield; every payout traces to a specific trade that already happened.
4. **Burn is enforced at lock time, not promised for later.** Half of every locked amount is gone the instant activation lands.
5. **Exits are never gated.** Unlocking $BULL deactivates the tier immediately, requires no one's cooperation, and can never carry a fee.
6. **Every distribution is a public, re-runnable computation.** The epoch snapshot, the tier weights, and the payout math are all data anyone can pull and recompute independently.

3. The link model

A link claim is one struct, signed twice under two different schemes because it spans two chains that don't share one:

```
struct LinkClaim {
    bytes32 solanaOwner;   // ed25519 pubkey of the NFT holder, raw bytes
    bytes32 nftMint;       // the Black Bull's mint address
    address evmWallet;     // the Robinhood Chain wallet being linked
    uint64  linkEpoch;     // increments to atomically replace a stale link
    uint40  expiry;        // claim is void after this timestamp
    uint256 nonce;         // holder-chosen salt: same terms, new nonce, new claim
}
```

`solanaOwner` signs the claim hash with a standard Solana wallet signature (ed25519). `evmWallet` signs the same hash under EIP-712. Both signatures are submitted to the attestor together; neither alone is sufficient, and a claim with only one valid signature is discarded, not partially accepted.

The attestor does not gate activation directly. Verified claims accumulate into a Merkle tree; at the close of each epoch, the attestor posts only the tree's root to a `LinkRegistry` contract on Robinhood Chain, tagged with that epoch number. The full claim set behind every root is published alongside it, so a link's inclusion, or its absence, is independently checkable by anyone who wants to recompute the tree from public data, not merely asserted by the service that built it. `activate()` (§4) takes a Merkle proof against the current epoch's root, so the lock contract itself never trusts the attestor's live say-so, only a specific committed root plus a proof against it.

> **Trust boundary, stated plainly.** *This is the one place in the design where Overcharge asks for trust rather than proving it outright: the attestor decides what enters the tree for a given epoch. A withheld claim delays activation by at most one epoch. A fabricated claim is not possible without a valid dual signature, since nothing enters the tree without one. §6 bounds the damage; §7 describes the path to removing this party entirely.*

4. Activation and tiers

```
struct Activation {
    address evmWallet;
    bytes32 nftMint;
    uint128 locked;        // remaining locked balance, post-burn
    uint8   tier;          // 0 = inactive, 1-5 = active
    uint64  sinceEpoch;    // epoch this activation last changed
}
```

Tier thresholds are set as a share of circulating $BULL supply, recalculated at each epoch boundary, rather than fixed absolute amounts. A fixed threshold drifts meaningless as the burn mechanic permanently reduces supply over time; a share-of-supply threshold stays the same commitment in real terms whether it's epoch one or epoch two hundred.

Tier	Lock threshold	Reward weight
I	0.02% of circulating supply	1.00x
II	0.05% of circulating supply	1.35x
III	0.125% of circulating supply	1.85x
IV	0.25% of circulating supply	2.50x
V	0.50% of circulating supply	3.33x

4.1 Lock

`activate(amount, merkleProof)` burns `floor(amount × 5000 / 10000)` immediately and credits the remainder to `locked`. Tier is recomputed against the threshold table snapshotted at the last epoch boundary, not against live circulating supply, so a burn landing mid-epoch elsewhere cannot move a wallet's tier out from under it before the wallet's own next snapshot.

4.2 Unlock

`deactivate(amount)` returns any or all of the remaining, unburned `locked` balance to the wallet on demand. Tier recomputes immediately and may drop to zero. There is no penalty beyond the burn that already happened at lock time, no cooldown, and no fee.

5. Epoch and distribution

Distributions run on a fixed seven-day epoch. At the boundary, the contract snapshots every activation's locked balance and tier. The epoch's reward pool is computed against this snapshot, taken at the epoch's start, never against balances at payout time. A wallet that locks after the snapshot earns nothing until the following epoch; a wallet that unlocks after the snapshot still receives what its snapshotted tier earned. This closes the standard reward-gaming pattern of locking immediately before a payout and withdrawing immediately after: the snapshot is what a payout is computed against, and it is already fixed by the time anyone could react to it.

```
walletShare  = (lockedAtSnapshot × tierWeight) / Σ(lockedAtSnapshot × tierWeight,
all active wallets)
walletReward = walletShare × epochRewardPool
```

PONS pays the treasury its 70% trading-fee share directly in WETH per trade, so no fee-asset conversion step is required before it can be split. At epoch close, a permissionless call, callable by anyone, not only the treasury, reads the accumulated WETH balance and splits it between two flows, each trade floored against the $BULL/WETH pool's own time-weighted oracle price so a manipulated low price cannot be used to lowball either buy:

- **Flow A, reward pool.** Buys $BULL and funds the epoch's reward pool, paid out per the formula above.
- **Flow B, $ANSEM reinforcement.** Bridges to Solana via the same native bridging path Robinhood Wallet already supports, and buys $ANSEM into the collection's existing, already-public buyback wallet.

6. Threat model and limitations

- **Attestor inclusion trust.** The attestor chooses what enters a given epoch's Merkle root. A withheld valid claim delays activation by at most one epoch; a fabricated claim is not

possible without a genuine dual signature, and the full claim set behind every root is published for anyone to audit.

- **Snapshot boundary gaming.** Bounded to at most one epoch's worth of reward per attempt by the snapshot rule in §5, and the boundary timestamp is fixed on-chain, not chosen by any party after the fact.
- **Bridge dependency on Flow B.** A downed or costly Solana bridge delays the $ANSEM side of a distribution without affecting Flow A, since the two flows execute independently.
- **New contract surface.** The activation and distribution contracts hold real locked value and require a security review proportional to that before mainnet exposure. None has been performed as of this draft.
- **No emissions, no price promise.** Every number in this document describes a mechanism, not a return. Reward size depends entirely on realized $BULL trading volume, which is unknown before launch.
- **Operator powers, enumerated.** The distribution trigger is permissionless. Tier thresholds, epoch length, and the Flow A / Flow B split are owner-adjustable within bounds compiled into the contract. The owner cannot access a wallet's locked balance, redirect an individual reward, or bypass the Merkle proof requirement on activation.

7. Staged decentralization

> **Forward path, design commitments, not yet deployed.** *The mechanisms below describe where the link layer goes after launch, not what ships with it.*

- **v1, single attestor.** One service verifies dual signatures and posts epoch roots. Full claim data is published every epoch for independent audit. This is what launches.
- **v2, threshold attestation.** The attestor becomes an n-of-m signing set, so no single operator can unilaterally withhold or fabricate a claim's inclusion.
- **v3, native verification.** The attestor's signature check is replaced by an on-chain or zero-knowledge proof of the original ed25519 signature, a technique already in production use by zk-based Solana identity systems, removing the attestor as a trust party and leaving only the Merkle commitment as public record.

8. Related designs

Native token-bound accounts (StonkBrokers). No linking layer needed at all, since the NFT already has an EVM address; simpler, with a smaller trust surface, but unavailable to any NFT that is not itself an EVM asset, which rules it out for a Solana collection outright.

One-time cross-chain airdrops. The common workaround: snapshot holders once, distribute a token allocation once, done. No ongoing infrastructure, but no ongoing relationship either; the token and the NFT drift into two unrelated assets within weeks. Overcharge is deliberately the more expensive design because the payoff is a permanent tie between the two assets, not a single event.

Wrapped or bridged NFT activation. Escrow the original on Solana, mint a synthetic on Robinhood Chain, activate the synthetic. This gives the synthetic a native EVM identity and avoids a linking layer entirely, at the cost of bridge custody risk on an asset that never needed to change chains, and a wrapped state that can desync from who actually holds the real one.

9. Conclusion

Overcharge's bet is that a holder can prove which wallet is theirs across two chains more cheaply, and more honestly, than a bridge can move the NFT itself. The asset never leaves Solana. The link is a signed claim, checkable against a posted root, not a database entry taken on faith. Locking burns immediately, unlocking is never gated, and every payout is computed against a snapshot anyone can pull and recompute. What remains a trust assumption at launch, the attestor's inclusion decision, is bounded to one epoch of delay and has a published, staged path out. Nothing here asks a holder to believe a roadmap; it asks them to check a root.

This is a pre-launch draft describing a system under active design. Parameters, thresholds, and mechanisms may change before any contract deployment. Nothing in this document is investment advice.