A GUIDE FOR ALLOCATORS AND THEIR AUDITORS · TESTNET

INDEPENDENT VERIFICATION OF DIAMOND HANDS CARRY'S HYPERLIQUID ACTIVITY.

This guide is for allocators and auditors who want to check the vault's Hyperliquid activity themselves. The data is labelled **TESTNET**. Trading perps can lose money; please read the risk disclosure (perpbrain.com/legal/risk-disclosure).

> **WHAT YOU CAN CONFIRM, WITHOUT TRUSTING GVNR**
>
> You can confirm that the funding the vault booked equals the funding Hyperliquid paid or charged for the same windows; that a BTC-perp position and its fills exist on Hyperliquid for the vault's address; and that the published receipt has not been edited, even by a fraction of a cent.
>
> The BTC vaults that hold depositor bitcoin, and the USDC bridge that moves margin onto Hyperliquid, are reported by Diamond Hands directly. This check covers the Hyperliquid leg.

WHAT YOU ARE CHECKING

Each time the vault's keeper books funding, the vault emits a **FundingMeasured** event on HyperEVM. Perp Brain compares every one of those events with Hyperliquid's own `userFunding` record for the same account, window by window, and seals the result in a receipt you can rerun yourself.

BEFORE YOU START

You need two files: `receipt.json` and `inputs.json`. No account, wallet or API key is required.

Get them from the Diamond Hands case study (perpbrain.com/case-studies/diamond-hands/), or from the GVNR data room. In the data room, new receipts land in the **Reports** folder after each Monday check. This guide sits in **03 Diamond Hands Carry**.

An agent can run the same check from a single file, with no browser: perpbrain.com/verify-dh.md.

STEP 1 · DOWNLOAD THE RECEIPT

From the case study page, download both files (or take the latest pair from Reports):

FILE	WHAT IT IS	ADDRESS
`receipt.json`	The sealed result: windows, totals and hashes	`perpbrain.com/case-studies/diamond-hands/receipt.json`
`inputs.json`	The raw FundingMeasured events and Hyperliquid funding rows	`perpbrain.com/case-studies/diamond-hands/inputs.json`

The walk-through below uses the hourly receipt for **3 Nov 2026, 02:10 to 16 Nov 2026, 10:00 (Asia/Qatar)**, labelled TESTNET. Vault address: `0x5310AE7e24835cC6f7740db18ebbF9f3aB422554`.

STEP 2 · VERIFY IT ON PERPBRAIN.COM/VERIFY

The check runs in your browser. The files are not uploaded or stored.

01 Open the page

Go to perpbrain.com/verify/.

02 Drop both files

Drop `receipt.json` and `inputs.json` onto “Drop the receipt files here”, or click **Choose files** and select both. You can also paste each file into its labelled box. A funding-reconciliation receipt has no `manifest.json`; leave that box empty.

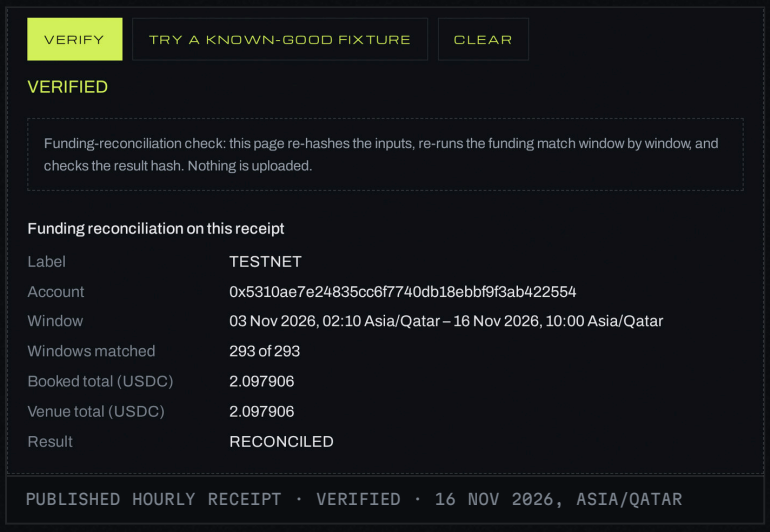
03 Run the check

The page usually starts as soon as the files are chosen. If it has not started, click **Verify**.

04 Read the result

A genuine copy of this receipt shows `VERIFIED`. The summary reports 293 of 293 windows matched, a booked total of 2.097906 USDC, a venue total of 2.097906 USDC, and result `RECONCILED`, labelled TESTNET.

To see a changed file fail, make a copy of `receipt.json` and change `measured_total_usdc` from `2.097906` to `2.197906`. Drop that copy with the original `inputs.json`. The page shows `MISMATCH` and lists `result`.





STEP 3 · CHECK THE SOURCE DATA ON HYPERLIQUID

A VERIFIED receipt means the math matches the saved inputs. This step checks those inputs against Hyperliquid itself. Open the receipt and note:

FIELD	WHAT TO COMPARE
`account`	The vault address, `0x5310AE7e24835cC6f7740db18ebbF9f3aB422554`
`window.start_ms` / `end_ms`	The period covered, also shown in Asia/Qatar on the verify page
Each window's `from_ms`, `to_ms`, `venue_usdc`, `venue_rows`, `measured_usdc`	The venue's funding rows in that window should sum to `venue_usdc`, and that figure should equal `measured_usdc`

Hyperliquid's public info API is a POST request, so it cannot be opened as a normal web link. For the last window of this receipt (16 Nov 2026, 08:55–10:00 Asia/Qatar) run:

```
curl -s https://api.hyperliquid-testnet.xyz/info \
  -H 'Content-Type: application/json' \
  -d '{"type":"userFunding","user":"0x5310AE7e24835cC6f7740db18ebbF9f3aB422554",
       "startTime":1794808515000,"endTime":1794812400066}'
```

You should see two funding rows whose `delta.usdc` values, `-0.291923` and `-0.396816`, sum to `-0.688739` — the `venue_usdc` and `measured_usdc` on that window. You can also look the account up in the Hyperliquid testnet explorer: app.hyperliquid-testnet.xyz/explorer/address/0x5310AE7e24835cC6f7740db18ebbF9f3aB422554.

To confirm a position and fills exist for the same address (these are looked up on the venue; they are not part of the funding receipt itself):

```
curl -s https://api.hyperliquid-testnet.xyz/info \
  -H 'Content-Type: application/json' \
  -d '{"type":"clearinghouseState","user":"0x5310AE7e24835cC6f7740db18ebbF9f3aB422554"}'

curl -s https://api.hyperliquid-testnet.xyz/info \
  -H 'Content-Type: application/json' \
  -d '{"type":"userFillsByTime","user":"0x5310AE7e24835cC6f7740db18ebbF9f3aB422554",
       "startTime":1793661022000,"endTime":1794812400066}'
```

In the first reply, look at `assetPositions` for a BTC position. In the second, each fill has a time, coin, price, size, side and hash. Hyperliquid keeps hourly funding rows for about a week and then rolls older rows into daily totals, so a receipt older than that can still be hash-checked against its saved inputs, but you may not be able to re-fetch the original hourly rows. That is why a fresh receipt is published each week.

STEP 4 · CHECK THE VAULT'S RECORDS ON HYPEREVM

Each window on the receipt lists a transaction hash and a block number. Open the vault on the HyperEVM testnet explorer (chain 998):

testnet.hyperevm-explorer.xyz/address/0x5310AE7e24835cC6f7740db18ebbF9f3aB422554. The Logs tab lists the vault's events. For any window, open its `tx` hash. The first window of this receipt is transaction 0xf5e083c9...d7e6: block `68590946`, 3 Nov 2026, 03:15 Asia/Qatar, success, sent to the vault.

The contract is not verified on the explorer, so `FundingMeasured` may appear as a raw log. Look for a topic that starts with `0x824d17d91917a8ddd61381e7656987988f2412d13970f80e04f611ec95a99445`. The first 32-byte word of that log's data is the measured amount in micro-USDC: `4800`, which is `0.0048` USDC, matching `measured_usdc` on the receipt.

READING THE RESULT

VERIFIED

The receipt is consistent with its inputs. The funding match was rerun in your browser (or in the Python check) and agrees, so the file has not been edited.

MISMATCH

The file does not match its inputs, or the inputs do not produce the stated result. The page lists the field that failed, for example `result`

when a total was changed.

MATCH · RECONCILED

For each window, and for the period as a whole, the amount the vault booked equals Hyperliquid's funding for that window, within 0.00001 USDC.

DIFF · DIFFERENCES

At least one window's booked amount does not equal the venue's. Those windows are the ones to raise with the vault operator.

Receipts older than about seven days. Hyperliquid keeps hourly funding rows for about a week, then rolls them into daily totals. You can still verify an older receipt against the inputs saved with it. You may not be able to re-fetch the original hourly rows from the venue. Use the daily running totals on the case study (`daily-check.json`) for the earlier period, and rely on the weekly receipts so the hourly record is not lost.

What a keeper gap looks like. It is a window much longer than an hour, still marked MATCH, because the next keeper call booked the whole gap in one go. In this receipt, 8 Nov 2026, 05:25–17:40 Asia/Qatar is a 12.25-hour window with 12 venue rows and a booked amount of `-0.410189` USDC, matching the venue. An earlier quiet period, from 2 Oct to 28 Oct 2026, sits in the daily running totals rather than this hourly receipt. In both cases the next entry caught up the amount in full; nothing was dropped.

FOR DEVELOPERS

The same full check, including rerunning the funding match, is one command. Put `receipt.json` and `inputs.json` in one folder; it prints `VERIFIED` and a summary line (293 of 293 windows for this receipt).

```
pip install "perpbrain>=0.1.4"
perpbrain verify-receipt receipt.json
```

You can also confirm the inputs fingerprint with no Perp Brain code at all. It must equal `inputs_sha256` on the receipt (`a7ce8a80750d0d06e68e96df66cb989b9100f5c6347e65e38680f9f8d8d3b213` for this file):

```
sha256sum inputs.json
```

From 0.1.4, `perpbrain verify-receipt` and the TypeScript helper `verifyReceiptAt` in `@perpbrain/sdk` recognise funding-reconciliation receipts by their `kind`. Version 0.1.3 and earlier print `kind mismatch` for this file; upgrade, or use the website.

The same instructions, written so an agent can follow them, are at perpbrain.com/verify-dh.md.

WHO TO CONTACT

Questions about a receipt or this guide: info@perpbrain.com.

Perp Brain is a GVNR product, owned by GVNR DAO LLC, a Marshall Islands DAO LLC. This walk-through uses TESTNET data. Perp Brain is developer tooling, and simulated results are not a forecast; trading perps can lose money, so please read the risk disclosure (perpbrain.com/legal/risk-disclosure).