ASTRYUM · PRODUCT DECK · SEPTEMBER 2026

Managed vaults

Delegate the decision, not the possession. A manager whose credential is verified on the ledger directs your capital inside a contract that cannot pay principal to anyone — and you leave whenever you want.

LIVE cage v2 on Flare mainnet · full cycle proven · surface “Managed vaults — run by a third party”

THE PROBLEM

Delegating management today means one of two bad trades.

In traditional finance: you know who the manager is, and you give up possession

The licence is real, the custody is theirs. When you may leave, and with what, is their decision and their books.

In DeFi: the code binds the manager, and nobody knows who they are

Curated vaults already stop a manager from extracting. But *who* the manager is, whether they hold any title, and until when, lives off-chain — in a website, a forum post, a name.

Astryum puts both halves together: **a credential on the XRP Ledger says who may direct, and a contract on Flare says what they can never do.** The client signs their own exit. Nobody gates it.

WHAT IT IS

Two users, one contract, no custodian.

The manager

Opens their vault (“cage”) from their own Xaman. Their root account carries credentials issued by third parties — a sector credential and an identity credential — with expiry. They pick venues from a governed list and **direct** capital: send it to work, recall it. They see a desk with orders and the status of their credentials.

The client

Deposits with their own signature into a pote (an ERC-4626 vault) and receives shares in their own name. Sees their position and an exit button. **Redeems with their own signature, whenever they want, without asking anyone** — the contract unwinds venues by itself if needed.

The manager can

Direct capital to a listed venue. Recall it. Set a payee for their cut, under a cap. Cede the seat temporarily.

The manager cannot

Extract. Send to an address. Touch shares. Block an exit. Raise the cap. Change the rules.

Astryum never

Signs. Occupies the director’s seat. Issues the real licence. Asks the client to approve the operator over their shares.

THE CREDENTIAL GATE

Without valid credentials on the root, the cage is not born.

- **Where the credential lives:** on the manager's root account on XRPL — a fresh, virgin account (one council = one cage, forever). Never on the pote, the Personal Account or a hot key.
- **What it reads:** OR-groups — `AIFM|CASP` and `KYC|KYB`: *the licence of your sector, and your identity*. Each an XLS-70 object with a mandatory `Expiration`, accepted by the subject (the accept is the consent), from an issuer on the accepted list.
- **When it bites:** at cage creation and at every council order — except accepting a pote and the **exits** (`recall`, `evacuate`), which are never gated. Fail-closed: an unreadable ledger is a refusal, never a pass.
- **What revocation does:** the credential expires and is not re-issued. New orders stop. Client exits never do.

Who issues — never Astryum

Astryum chooses which issuers it accepts and checks the object on the ledger with one uniform rule. The notary robot re-checks facts anyone can re-check (today: an identity attestation from Coinbase on Base; a domain, its `toml` binding and a register entry for the sector credential) and issues with typed refusals — "could not read" is never "no".

Said plainly: in the rehearsal the sector credentials are demo issuance with the public register's link as URI. Astryum brings on chain what the register says; it certifies nothing. In public we say only *a credential verified on the ledger*.

BIRTH — RAIL A

One XRPL signature from the root deploys the cage and its first pote.

Manager's root (XRPL)
Credentials accepted. Constitution anchored with DIDSet. Signs a Payment to the FAssets Core Vault, memo = hash of the batch.

FDC attests · executor delivers →

Root's Personal Account
Derived from the XRPL address. Executes the batch.

only this PA may create →

AstryumCageFactory
Checks `getPersonalAccount(root) == msg.sender`. Deploys `XrplCouncilBridgeV2` (council hash, anchor hash) → cage → bind.

→

AstryumCage
`AUTHORITY` = that bridge, immutable. Creates the first pote through `PoteDeployer`.

→

Pote (AstryumVaultV2)
ERC-4626. Shares to depositors. Venues only from the registry.

Three potes free

From the fourth, a creation fee of 1 FXRP goes directly from the payer to the treasury inside the birth transaction. Immutable per cage.

The gate is in the way

The backend refuses to prepare the birth without valid credentials on the root — and the bridge v2 already verifies the anchor for the day the anchor becomes a door the ledger closes.

Status

LIVE factory, registry and anchor on mainnet since 6–8 Sep; three cages born so far.

ORDERS — RAIL B

Every manager order is a council order. The bridge verifies; the cage obeys or reverts.

Root
1-drop Payment to the v2 anchor rLcoFM… YDn7, memo = hash(nonce, calldata). Credential IDs can ride along.

FDC proof →

Bridge v2
Nine guards incl. `WrongAnchor` · consumed txId · sequential nonce

onlyAuthority →

Cage
`create-pote` · `direct-to` · `recall` · `move` · `evacuate` · `set-payees` · `cede` · `end-cession` · `accept-pote`

→

Pote → venue
Kinetic isoFXRP · Firelight stXRP — only if the registry approves that exact target and kind

anything else →

Revert
The ledger says no.

The gate, again, on every order

Same credential check as at birth — except exits. A manager with an expired credential can still `recall` and `evacuate`; they cannot direct.

Proven

LIVE Bridges of the three cages at nonces 3 / 1 / 3 (15 Sep). Full cycle — create pote, deposit, direct, recall, redeem — executed on mainnet on 8 Sep.

THE CONTRACT'S RULES

"The director is never approved" is not a rule. It is unrepresentable.

The pote (AstryumVault / V2)

- ✓ ERC-4626 with reentrancy guard; `totalAssets` is the accounting — no separate ledger of principal.
- ✓ `requestRedeem` is **owner-only, with no allowance path**. `claimRedeem` is permissionless, receiver fixed by the holder.
- ✓ Venues typed: ERC-4626, Compound-v2, queued ERC-4626. Adding one requires `REGISTRY.isApprovedVenue(chain, target, kind)`, code at the target, and the same underlying asset.
- ✓ Optional cooldown (0 = synchronous redeem; max 30 days) and a liquidity buffer floor (max 50 %).
- ✓ Payee cap set at birth by the cage; a pote asking above 20 % is born private with a permanent gate.
- ✓ **No protocol fee hook.** Astryum's fee to the client is zero, enforced by absence.

The cage (AstryumCage)

- ✗ No `receive`, no `fallback`, no `sweep`, no generic `call`, no function over shares — it custodies nothing, ever.
- ✗ No proxy, no pause, no owner = Astryum. Succession only with 30 days and five verified continuities; the payee cap cannot rise.

Invariant 18, with a tripwire

Shares inherit standard `approve/transferFrom`. A single approve from the client to the director would dissolve the cage without touching the contract. So: **no flow ever asks the client to approve the operator over their shares** — a test decodes every route's calls and fails if any approve targets anything but the pote itself.

THE REGISTRY — THE SCANNER, MADE A CONTRACT

Which venues a cage may reach is a governed list, not a setting.

- **AstryumRegistry** on Flare: listing waits a timelock (1 hour, immutable) and emits a public event; delisting is immediate. Governor in two steps.
- The pote itself consults the registry when a venue is added (`_addVenue`) — never Astryum's server.
- Honest limit, written in the contract: the registry is consulted on *add*, not on *allocate* — a delisting does not pull capital already inside. Recall does.
- Live today: **2 venues** — Kinetic isoFXRP (Compound-v2 kind) and Firelight stXRP (queued ERC-4626 kind), active since 6 Sep.

The catalogue is exhausted — and we say so

A new venue must be a typed shape, on the same chain, with the same underlying asset. Upshift-style vaults (earnXRP, Monarq) hold shares in a separate token and are not ERC-4626; Sceptre's asset is FLR. **Zero new venues fit a pote today without new Solidity** — a branch per venue class, audited. Deployed potes cannot adopt a new kind (no proxy): new kinds mean new potes; migrating is redeem and re-deposit.

Open blocker: the registry's governor is the treasury/executor key. Moving it to a multisig is first in the next wave.

THE WHOLE STRUCTURE

Issuer → root → factory → cage → pote → venues. The client talks only to the pote.

Issuer
accredited third party: CredentialCreate

subject accepts →

Manager root (XRPL)
credentials · DIDSet · signs birth and orders

rail A · rail B →

Bridge v2 → Cage
AUTHORITY
immutable · cannot custody

→

Pote (ERC-4626)
shares · payee cap · redeem owner-only

registry-approved only →

Venues
Kinetic · Firelight

Client

Deposits and redeems with their own signature, directly against the pote. Never sees the manager's keys, never approves the manager.

XRPL

virgin root · Credentials XLS-70 · DIDSet

Payment + memo · optional SignerList

FLARE

FDC · Smart Accounts · FAssets FXRP

CageFactory · Cage · Registry · PoteDeployer · VaultV2 · Bridge v2

LEGAL VEHICLE

A company that manages third-party capital — and the personal variant of the same pote.

The manager as an enterprise account

- One root on XRPL per legal vehicle; the board can be the root's SignerList.
- Credentials the law already requires, translated to XLS-70: **sector licence + company identity** (or personal identity for an individual manager). Issued by the register, the regulator's attestation, a trust-service provider — never by Astryum.
- By-laws anchored (DIDSet). The root is the vehicle's book of acts: hash + pointer, never content.
- Gating the root gates the whole tree: cage, potes, subordinate accounts obey by binding and carry no credential of their own.
- Copy rule: *a credential verified on the ledger*. Never "licensed", "regulated", or the name of any regulation.

Personal managed vault — manager == user NEXT

The same pote with yourself as manager: you set your limits today and the code enforces them tomorrow, even against you at 3 a.m. — the Ulysses contract. Managing your *own* capital is not managing third-party capital, so the credential is identity, not sector. Depositors limited to your own verified wallets. Mechanics exist; the surface comes next.

The agent's sandbox LATER: a cage admits three free potes; pote B as the compartment of an agent you designate — its limit is its pote, revoking the designation switches it off, your exit never depends on it.

LIVE ON MAINNET

Deployed 3–8 September. Full cycle proven. Read on 15 September.

3 · 4

cages · potes born on the factory (one of them the exchange's); all fully redeemed today

2

venues live in the registry: Kinetic isoFXRP, Firelight stXRP · timelock 1 h

168 / 168

Foundry tests green · 62 on the cage v2 against real venues on a mainnet fork

1

theft reproduced in v1 and blocked in v2, kept as a permanent test

The cycle, on mainnet (8 Sep)

Create pote by council order → client deposit → direct to venue by council order → recall by council order → client redeem. Manager gate fail-closed at birth with credentials in force on the root (identity from a real attestation on 9 Sep).

What is founders-only today

The creation tunnel for non-founders stays behind the founders-only wrapper; the manager's desk and the public directory are published. Own-capital caps are small and written down: no external audit yet.

COMPOSITION IS NOT AUTONOMY

A person decides. The code decides what they can’t.

Two axes, not one

Composition is who designs the allocation (manual → composed → assisted). **Autonomy** is who triggers and with what authority. We climb in composition without climbing one step in autonomy: however elaborate the plan, it ends the same way — the manager signs, and the contract imposes the limits (cap per venue, liquid floor, allow-list with delay, an exit nobody gates).

Never sold as autonomous

There is a person deciding, and that honesty is the advantage. Astryum compiles and explains; it never proposes “for you” — a personalized proposal is advice, and advice belongs to an occupant with a credential, never to the tool. The system does not set weights the user did not state.

The sector says the same thing without knowing it: “designing risk policy and constraints which models, agents or autonomous systems execute against” — that is the cage, stated by others. Our position is the split of layers: the venue is a line in the allow-list, not the rail; today a person with a credential checked on the ledger decides; tomorrow a program may — and the limits and the exit do not move.

ECONOMICS

The client receives no less because Astryum is in the middle.

The manager’s cut

A payee on chain, on **yield only**, never principal. Ceiling set by the cage at birth: `MAX_PAYEE_BPS_ALLOWED = 2000` — 20 % — and succession cannot raise it. Above 20 % a pote is born private with a permanent gate.

Astryum’s fee to the client

Zero, enforced by absence: the pote has no fee hook. The written canon: *Astryum charges the operator, never a cut of the client’s yield.*

What Astryum charges (next)

A creation fee from the fourth pote (1 FXRP, live). Transport at cost per order (built, off). The principal source, decided and not built: a **distribution fee paid by the venue** on the capital that reaches it — same rate for the class, never affecting order, always disclosed.

Never: a percentage of shares or of client yield · prepaid balances · a token · a charge in a flow that does not pass through a registered regulated partner. Today nothing is charged.

TRUST MODEL

Who can harm whom — and what stops them.

Can the manager rob the client?

They cannot extract: two verbs exist — send to a listed venue, recall. No “send to an address”. The list is a registry with a timelock and a public event. What they *can* do, and we say it: choose badly within the list.

Can Astryum?

No, by absence of capability. No key of ours sits on any cage, pote or root. The relayer carries bytes the manager committed to. The contracts have no proxy, no pause, no owner we could use.

If Astryum disappears?

The manager orders from any XRPL client; the client redeems from any EVM client; the credentials live on the manager’s own account. The only thing anyone loses is convenience.

What we tell a reviewer before they find it

Registry governor = treasury/executor key (multisig next). The on-ledger credential door is built and not switched on — today the backend gates, then the ledger will. The risk scanner does not cover Flare and passes “unknown” — a chain guard is first in the urgent list. No external audit: own capital, small caps.

NEXT AND AFTER

A third generation of the contract, an audit, and a machine that certifies venues.

NEXT Waves 0–3

- Registry key to a multisig; scanner chain guard; credential renewal without a gap.
- **Verification machine:** one venue class declared once; a fork battery for Kinetic and Firelight producing a dated certificate; a venue card the explainer reads; a filter by exit time.
- **Enterprise account:** KYB provider under contract; anchor door switched on in mainnet; the kit (API keys + remote MCP).
- **Contract v3:** a branch per venue class (queued venues first) + **external audit** — the number-one blocker, the main line of the grant applications.
- Distribution fee with attribution and disclosure.

LATER

- The agent's pote: a designation the root can revoke; permission delegation (XLS-75) as transport when it activates — never the security boundary.
- Protocol-managed wallets as the cage's remote arm: capital in wallets the cage governs on other chains. Frame exists; the arm reverts on purpose.
- New potes for new assets (FBTC when it reaches Flare mainnet): the asset is immutable in the constructor — a new pote, not a new venue.

EVIDENCE

Read it on flarescan and xrpscan.

Flare (chain 14)

AstryumRegistry 0x18c807d107EF9d337aFA5BdD39b190C42Dc50Aec
AstryumCageFactory 0xE897fFef10F950Abc2d9c759107410F713e941D9
PoteDeployer 0x5f84c5F4805c29298AF71Fc6657D3E780eF27E9E
Kinetic isoFXRP 0xD1b7A5eFa9bd88F291F7A4563a8f6185c0249CB3
Firelight stXRP 0x4C18Ff3C89632c3Dd62E796c0aFA5c07c4c1B2b3

XRP Ledger

v2 order anchor rLcoFM9XF8CL5GoFyMACguhdnDtwkNYDn7
Notary issuer rHKxjrGRrCegQhLrdnXEPAeGyeJ1JR4Hae
Source Tag 2607090002

Contracts: contracts/src/{AstryumCage, AstryumCageFactory, AstryumRegistry, AstryumVaultV2, XrplCouncilBridgeV2}.sol · gate: ManagerCredentialGate.ts

Product: astryum.xyz/app/manager · /proof · code: GitHub.

ASTRYUM MANAGED VAULTS

Delegate the decision. Keep the possession.

A credential on the XRP Ledger says who may direct. A contract on Flare says what they can never do. You sign your own exit.