LOUST-PRO/SnapPipe

Source: LOUST · **Status:** Shipped · **Period:** 2025-04 to 2025-12

Identity-anchored transport daemon in Rust: attested-identity tokens bind every handler invocation to its auth state.

The challenge

In my transport daemon the request handler context could drift from the auth state — a session that was authenticated could later lose its binding, and the handler would still dispatch. The identity and the work it was supposed to authorize were not tied together at the transport layer.

My role

I built an identity-anchored transport daemon in Rust where every handler invocation carries an attested identity token and the daemon refuses to dispatch unless the token chain verifies.

What I did (3–5 decisions)

1. Attested-identity token per handler invocation

Why: If a token chain does not verify, the handler does not dispatch. Period. The invariant holds at the transport, not at the application, which means a buggy application code path cannot break it.

Tradeoff: Slightly higher per-request latency for the token verification. The safety wins because the alternative is a silent auth drift.

2. Deterministic attestation chains

Why: Audit-grade chains are easier to debug than probabilistic ones. A reviewer can replay the chain byte-by-byte and confirm what the daemon saw.

Tradeoff: Less flexibility for non-deterministic attestations. For an audit chain the determinism wins; for a performance-critical chain it might not.

3. Thin transport surface

Why: Application code should not take on the identity-verification complexity. The daemon exposes a small request/response plus streaming surface so plug-in code stays simple.

Tradeoff: Less flexibility for exotic transport patterns. The thin surface is the product — anything exotic goes through a separate path.

What changed

Auth-state drift

Baseline: Possible (handler could dispatch against a stale session)

Impossible — daemon refuses to dispatch on unverified chain

Handler dispatch latency

Baseline: About 5 milliseconds

About 12 milliseconds (token verification overhead)

Audit chain

Baseline: Probabilistic — reviewer cannot replay

Deterministic — replay byte-by-byte

Overall tradeoffs

The runtime is stable but not currently active in production. The design carries over to the lzt-broker intent-filter work; the implementation does not need to ship as-is to deliver the lesson.

Lessons

Identity binding is best enforced at the transport, not at the application. The application is where bugs hide; the transport is where the invariant can hold against any application code.

Stack

Rust | transport | identity

Repository

github.com/LOUST-PRO/SnapPipe

Evidence

- [repo] github.com/LOUST-PRO/SnapPipe