INTERNAL TOOLING

A retried order cannot become a second order

A paper trading lab, simulated money only, with stable order ids, a pre-trade risk gateway and a hash-chained audit log.

Sytze P.
Solution Engineer

The problem

Most of the cost in a system where mistakes are expensive sits outside the feature. A request times out after the other side already accepted it. A limit exists only in someone's head. A log cannot prove what actually happened.

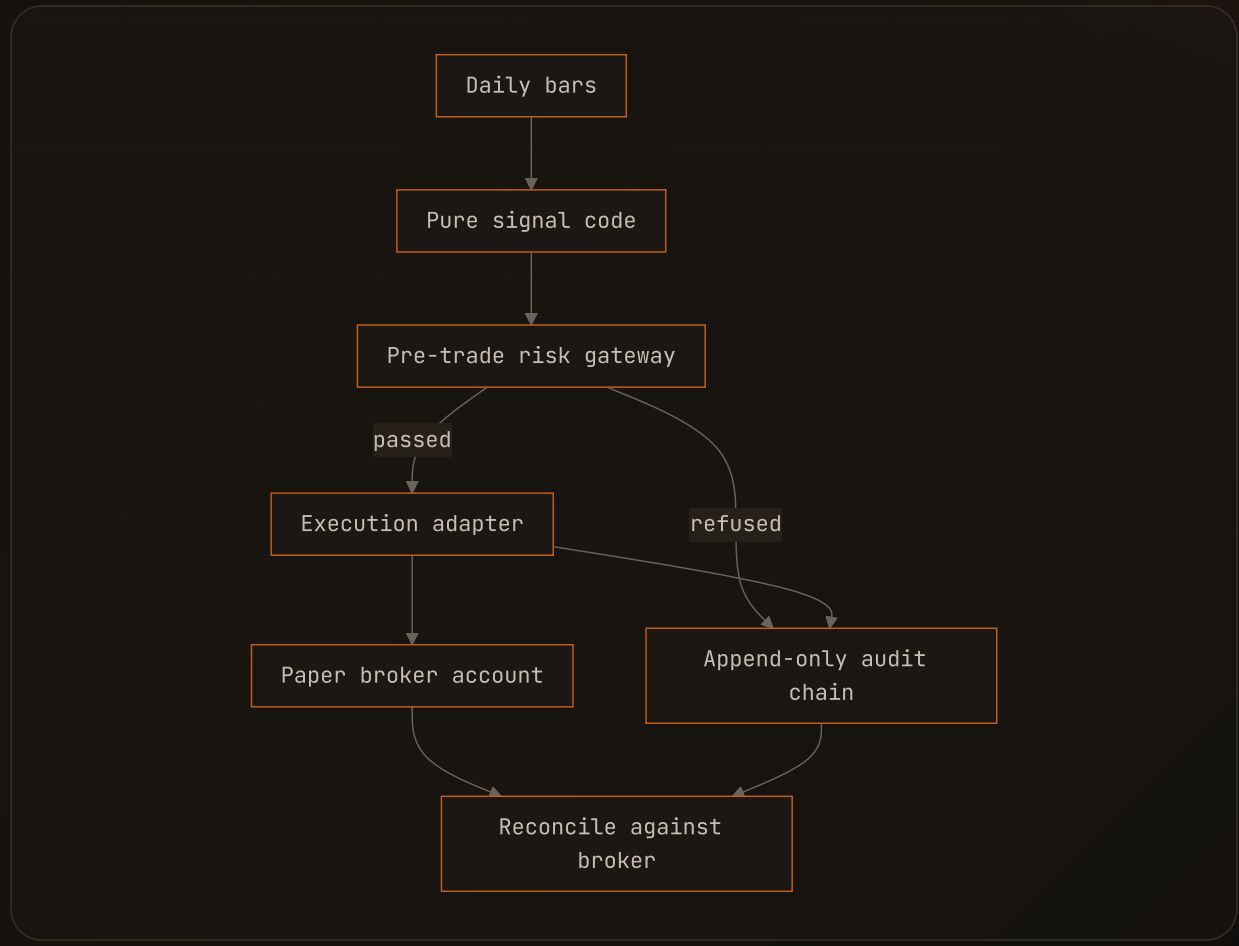
This is a personal project built to work on exactly that, against a simulated broker account. Nothing here trades real money, which is the point. A paper account is the cheapest place to make the expensive mistakes.

APPROACH

Signal to audited order

Four seams, each a module that could become its own process later. Signal code is pure and may not import the broker. Risk is a separate gateway that validates an order or refuses it. Execution is the only place the broker SDK appears. The audit log is append-only and hash-chained, so an edit to any earlier entry breaks every hash after it.

One rule carries most of the weight. Every order id is derived from the signal that produced it, so a retry after a lost acknowledgement lands on the same broker-side dedup key instead of placing a second order.



HIGHLIGHTS

How it works

SAME BAR, SAME ORDER ID

The id comes from the signal, anchored to the bar's own timestamp rather than the wall clock. A re-run of the same bar reaches the same broker-side dedup key instead of filling twice.

REJECTIONS AND TIMEOUTS ARE OPPOSITES

Transport errors back off with full jitter up to a capped attempt count. A broker rejection is terminal, and a programming error fails fast rather than being retried as flaky network.

LIMITS FAIL CLOSED

Order caps, an exposure cap and a daily-loss breaker gate entries only, so an exit is never blocked. An unreadable equity anchor trips the breaker instead of quietly disabling it.

THE LOG HAS TO AGREE WITH THE BROKER

One command walks the hash chain and checks every link. Another diffs broker orders, fills and positions against the log, then exits non-zero on a break so a scheduler can react.

RESULTS

Results & impact

same signal, same order id

a retry cannot become a second order

exits are never blocked

caps and the loss breaker gate entries only

log diffed against the broker

reconcile exits non-zero on a break

STACK

Tech stack

Python
pipeline and CLI

alpaca-py
broker SDK, paper account

pandas
bar data and indicators

Click
commands with ops exit codes

pytest
unit and CLI tests

mypy
type gate, enforced in CI

Sytze P.

Solution Engineer

A paper trading lab, simulated money only, with stable order ids, a pre-trade risk gateway and a hash-chained audit log.