# lzt-harness

**Source:** Private · **Status:** Active · **Period:** 2025-02 (active)

Private operator harness for Claude Code: multi-agent routing, destructive-command hooks, and an intent-filtered MCP broker.

## The challenge

A single-agent Claude Code session loses context faster than I can ship useful work, and the failure modes of unattended agentic execution (silent git operations, accidental file deletion, runaway tool calls) need belt-and-suspenders enforcement rather than a hopeful prompt.

## My role

I built a multi-agent harness on top of Claude Code with planner, implementer, and reviewer routing, hooks that block destructive operations by default, and an MCP broker that surfaces intent in response metadata.

## What I did (3–5 decisions)

### 1. Multi-agent routing tier

**Why:** Specialization beats raw context. A focused implementer does not drift the way a single agent juggling planning and execution does, and a reviewer catches what the implementer misses.

**Tradeoff:** Higher coordination overhead per task and longer time-to-first-output for trivial work. The discipline is worth the overhead for any task that crosses a single session.

### 2. Hooks that block destructive operations by default

**Why:** Belt-and-suspenders enforcement: the agent should never be one keystroke away from wiping a working tree or force-pushing a branch. The hooks block by default and require explicit operator approval to bypass.

**Tradeoff:** About 50 milliseconds of latency per tool call. Bypass requires a literal approval phrase in the same turn, which slows me down too — by design.

### 3. Intent-filtered MCP broker

**Why:** Only allow-listed tools fire telemetry, and the broker surfaces what an agent asked for versus what it actually got back. The audit chain stays readable even when a session runs for hours across multiple sub-agents.

**Tradeoff:** New MCP integrations require an explicit allow-list entry. The friction is intentional — the broker is the chokepoint that keeps unexpected tools from running silent.

## What changed

**Effective session length**

Baseline: About 30 minutes per single-agent session

4–6 hours across multi-agent sessions

Evidence: Internal session logs, 2026-Q3

**Silent work-loss incidents**

Baseline: About 2 per month

0 in 2026-Q3 (remaining cases caught by the teleport safety net as advisories)

**Audit chain readability**

Baseline: Opaque — single agent, no per-tool trace

Per-tool intent log, browsable after the session

## Overall tradeoffs

The harness is private because the rules and hooks are tuned to my own failure modes. The patterns (multi-agent routing, destructive-command hooks, intent-filtered broker) generalize, but the specific rule files do not.

## Lessons

The failure mode of agentic work is not the agent — it is the missing guardrails around it. A disciplined harness turns the agent from a liability into a tool I trust at 3am.

## Stack

Go | Rust | Python | Shell | MCP

## Evidence

- [repo] github.com/LOUST-PRO