SANDBOXED WORKTREES · AI-AWARE VERSIONING · PERSISTENT MEMORY

Auditable Workspaces for AI Coding Agents.

- h5i gives every coding agent a sandboxed Git worktree.
- h5i records the evidence behind each change: prompts, commands, logs, policies, and reviews.
- In Git, no SaaS.

★ 450+ GitHub stars · the #1 worktree manager by stars

WHY NOW

AI agents are becoming production labor.

TODAY

An agent run is unsafe and opaque

- ✕ Untracked prompts and context
- ✕ Dangerous tool usage
- ✕ Conflicting multi-agents

WITH H5I

An agent run is safe and auditable

- ✓ Sandboxed environment
- ✓ Tracked prompt and context
- ✓ Conflict-free multi-agent development

THE PROBLEM

The pain is already here.

PROMPTS

Untracked prompts

Can't improve them or reproduce a result.

CONTEXT

Lost context

Every session re-grounds from scratch.

SAFETY

Unsupervised tool use

Wrecked envs, malware, secret exfil.

TOKENS

Multi-agent token waste

Each agent re-reads the whole repo, and run many tools.

CONFLICT

Multi-agent conflict

They overwrite each other's work.

AUDIT

Hard to audit

Which prompt or model shipped a change? Unknown.

THE SOLUTION

h5i solves all six.

PROMPTS

Prompt versioning

Captured with each commit, so results replay.

CONTEXT

Persistent context

Carried across sessions, no re-grounding.

SAFETY

Supervised sandbox

A policy-enforced worktree per agent.

TOKENS

Token reduction

Raw output out of context, up to 95% less.

CONFLICT

Conflict-free ensemble

Isolated worktrees, merged into one result.

AUDIT

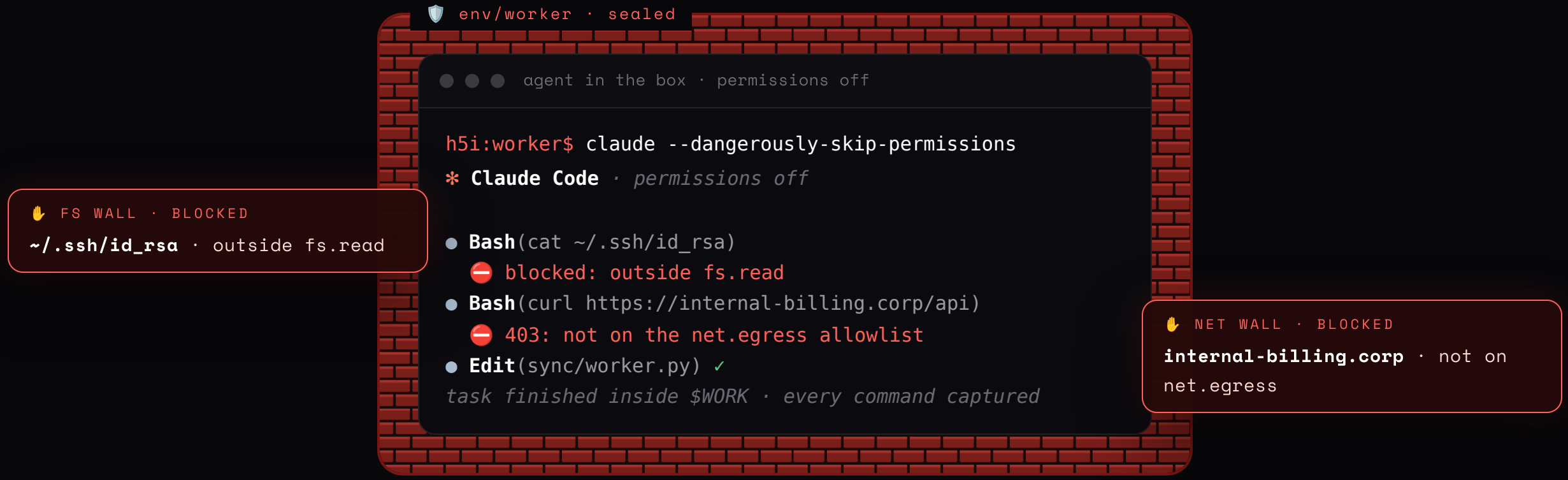
Easy to audit

Risk-ranked review of every AI change.

THE SANDBOX

A sealed workspace per agent.

Turn agent permission prompts off: the box is the permission. Writes stay in `$WORK`, network egress is allowlisted, and every denial is logged as evidence.



The agent **still finishes the task**. Only the reviewed diff is merged.

THE RECEIPT

Every diff gets a receipt.

The prompt, the reasoning, the commands, the tests, the denials: attached to the commit, versioned in Git, recallable from any clone. No SaaS.

a week later · any clone

```
$ h5i recall log --limit 1
commit 4f2c9e1  fix: clamp retry jitter
  agent     claude-code · claude-opus-4-8
  prompt    "Fix the flaky retry test in tests/test_worker.py"
  context   goal + reasoning trace, restorable per commit
  tests     241 passed · coverage +1
  denials   2 blocked · logged as evidence
  tokens    -94% (412 → 9 lines, raw recoverable)
```

- ✓ Prompt
- ✓ Reasoning context
- ✓ Commands
- ✓ Test output
- ✓ Denied actions
- ✓ Final diff

A week later, you can still answer: **why did this diff happen, and was it built safely?**

AUDIT

Triage the riskiest AI changes before they merge.

Receipts make triage automatic: `h5i audit review` ranks commits by uncertainty, blind edits, churn, and scope, so review starts with the ones that need it most.

h5i audit review

```
$ h5i audit review --limit 200
Suggested Review Points — 2 of 200 commits flagged

  #1 a3f1c0d  score 0.94  ████████
     auth: refresh token rotation
     ▸ security-sensitive · API token leakage · no tests

  #2 92c80f4  score 0.71  ███████
     db: migration for sessions table
     ▸ large diff · broad scope · low prompt certainty
```

TOKEN REDUCTION

Tool output is trimmed before it burns through the context window.

The receipt never bloats the context: `h5i capture run` keeps every command's raw output out-of-band and hands the agent a compact, structured summary. Nothing is lost, only the noise.

RAW

every line into context

```
$ pytest -q
tests/test_api.py ..................  [ 18%]
tests/test_auth.py ...F............... [ 41%]
tests/test_cache.py .................   [ 60%]
tests/test_db.py ....................   [ 82%]
tests/test_util.py ..................   [100%]
  … 412 lines: every PASS, warning, and
    traceback frame streamed into context …
========== 1 failed, 240 passed ==========
```

≈ 4,600 tokens spent

⇒

FILTERED

h5i capture run · structured

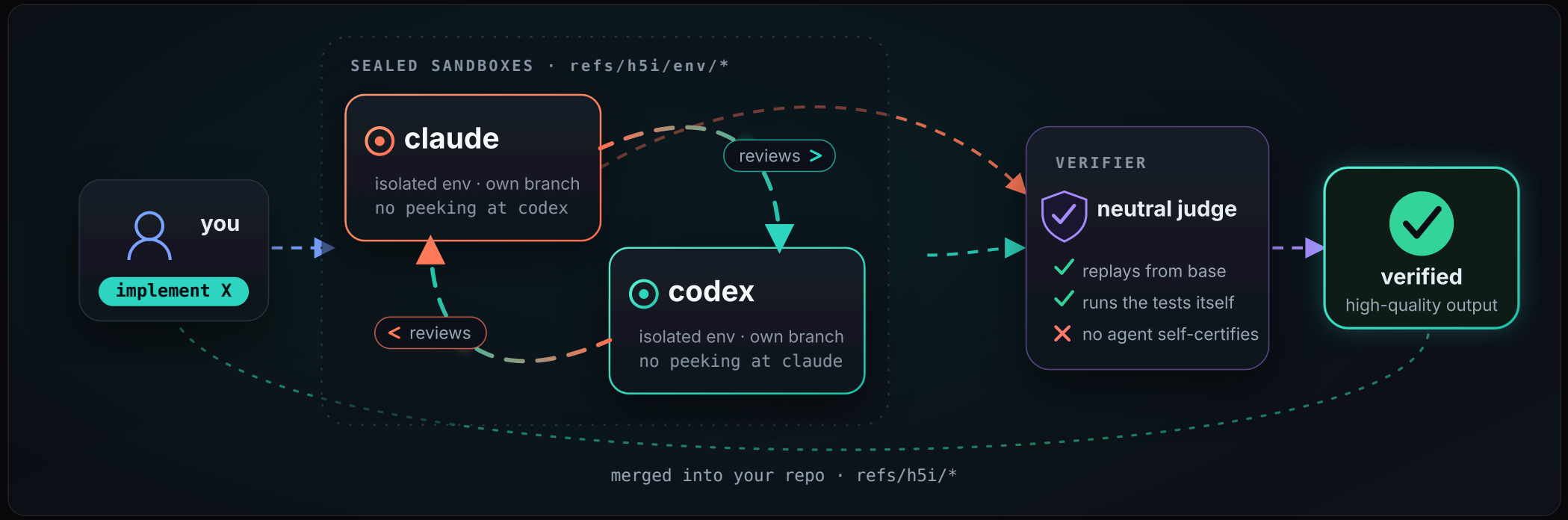
```
$ h5i capture run -- pytest -q

  1 failed · 240 passed  (240 ok collapsed)
  ▸ tests/test_auth.py:88  expected 200, got 401
```

≈ 280 tokens · 94% fewer · rehydrate: h5i recall object 41a337be

THE TEAM

Scale to many with `h5i team`.

`h5i team` fans one task out to agents in sealed worktrees, so they never overwrite each other. They peer-review each other's work, and only the winner merges.



TRACTION

Agent-heavy developers are already pulling this.

450+	30+	8+	#1
GitHub stars	forks	active contributors	worktree manager by GitHub stars

BEACHHEAD

Teams that generate code faster than they can safely accept it.

We spoke with dozens of developers during PhD research in AI and systems security. The repeated bottleneck was not code generation. It was safe code acceptance.

BEACHHEAD · BUYS FIRST

Agent-heavy engineering

Already using Claude Code, Codex, Cursor, or internal agents, and drowning in unverifiable patches.

EXPANSION →

Platform / DevEx

A standard workflow for running and reviewing many agents.

Security

Sandboxed autonomy and evidence for every blocked action.

Governance & regulated

Fintech, health, government: audit every AI-written change.

WHY H5I WINS

h5i wins by becoming the default auditable workspace for agents.

Agents will write more code across more repos. Teams will not trust raw diffs alone.

Coding agents	generate patches.
Sandboxes	contain risky execution.
Observability	records what happened.
h5i	**gives every agent run a workspace with isolation, provenance, logs, policies, and reviews.**

The scarce resource is no longer code generation. It is safe code acceptance.

WHY NOW · WHY US

We will make h5i the default way teams accept AI-written code.

Why this exists

- PhD research in AI + systems security.
- Dozens of developer conversations on agent safety, sandboxing, and AI-code review.

Built already

- 450+ GitHub stars.
- #1 worktree manager by GitHub stars.
- Dogfooded with real agent-team workflows.

Next 12 weeks

- Ship the GitHub PR verifier.
- Convert 10 design partners.
- Run 1,000 verified agent attempts.
- Land first paid pilots.

Git tracks the **diff**. **h5i tracks the run.**

h5i.dev

github.com/h5i-dev/h5i