

The Record Does Not Change

What a verifiable record of AI-agent work costs, what it saves, and how it works, measured on one company since March 2026

AppendOnly by 5R Industries · Adrian Outlaw · full report (text version 11)

Abstract. Companies are spending on AI agents faster than they can tell whether the spend is working. Surveys through 2026 put enterprise generative-AI spend at tens of billions of dollars a year, while 95 percent of pilots show no measurable profit-and-loss effect, more than 40 percent of agentic AI projects are forecast to be cancelled by 2027, and a third of the time AI saves is lost to checking and redoing its output. The missing piece is not a better model. It is a record: proof of what an agent actually did, kept where the agent cannot quietly change it, checked by something the agent does not control, and readable by a person who does not work in AI. This report describes such a record, running in its first form on a small engineering company since March 2026 and in its present form since 9 September 2026, when signing and public timestamps began (action capture followed the next day); most of the numbers below come from the period since July, when the independent verifier started. The company is run by one person with an AI chief of staff. It gives the measured cost of running it, the five lines on an income statement where the money comes back, six industries where those lines are already regulated or already bleeding, and the honest boundary of what one deployment can and cannot prove: one operator, one company, one agent, and a claim-labelling precision checked so far on five decided rows, all of them rows the verifier had flagged. The technical design, chapter by chapter, is Part B; a case study of one afternoon's work run on the record is Part C.

About this release. This is the full report: the argument (Part A, sections 1 to 7), the system as deployed (Part B, sections 8 to 21), the measured results and related work (sections 22 to 27), and the case study of one afternoon's work run on the record (Part C, sections 28 to 35). It follows the first public release of 29 September 2026 (Part A and the results, text version 2), which is carried here unchanged except for three sentences listed in the corrections table; the case study (Part C) is published here for the first time. Every number of ours in Part A and in the results was measured on 17 September 2026 from the live system; Part B's sentences were checked against the record on 26 and 27 September 2026 and carry the counts of that check; Part C's numbers were measured on 26 September 2026 from its own chain; each is reproducible by us from files in the record, with the commands and their output committed in dated measurement files beside the release. What a stranger can check today is the integrity of the record (its signatures and public timestamps) and, from the evidence bundle published with the first release, that the history since 9 September 2026 was not rebuilt; not yet the record's contents. The filing that covers the mechanisms was made on 24 September 2026. This report was written on the record it describes: the agent that helped write it is the agent verified in it, and every draft is a signed commit in that record. This release is licensed CC BY 4.0.

Part A. The argument

1. The question every buyer is asking

A chief financial officer who approves an AI budget is asked, a year later, what it bought. In 2026 the honest answers in most companies are not good. The numbers below are other people's, gathered on 17 September 2026 and checked against their sources on 26 September 2026; every source is listed with its link in the Sources section at the end of this report.

- Enterprise generative-AI spend rose from \$11.5 billion in 2024 to \$37 billion in 2025 (Menlo Ventures). Enterprises expect their spend on large language models to rise about 65 percent in 2026 (a16z, January 2026), and by mid-2025 the share of that spend paid from innovation budgets had already fallen from a quarter to 7 percent: AI is now a core operating line, not an experiment (a16z, June 2025).
- 95 percent of enterprise generative-AI pilots show no measurable profit-and-loss impact (MIT NANDA, 300 deployments, 2025). The best rebuttal to that study is that most companies never took a baseline, so they could not have measured an effect either way. Both readings say the same thing to a CFO: nobody can show where the money went.
- Gartner forecasts that more than 40 percent of agentic AI projects will be cancelled by the end of 2027 for “escalating costs, unclear business value or inadequate risk controls,” and that by 2027, 40 percent of enterprises will pull autonomous agents back “due to governance gaps identified only after production incidents occur.”

- Only 26 percent of large U.S. companies report full, real-time visibility into what their AI systems cost to operate (KPMG, Q2 2026).

Two things are happening at once. The output cannot be trusted, so people check it, and the checking eats the saving. And when something goes wrong, there is no record of what the agent did, so nobody can find out.

The review tax. 37 percent of the time AI saves is lost to correcting and redoing its work, about four hours lost for every ten gained, and 77 percent of daily users say they review AI output as carefully as a colleague's, or more (Workday and Hanover Research, 3,200 respondents, January 2026). Workers spend about 47 days a year on AI and about 20 of them troubleshooting errors and re-prompting (BambooHR, September 2026). Substandard AI output costs the person who receives it about two hours per instance, about \$186 per employee per month, more than \$9 million a year for a 10,000-person company (Stanford and BetterUp, 2025). In software, teams using AI merge more code with more bugs and review it five times slower (Faros AI, 22,000 developers, 2026). Leading legal research tools gave incorrect or mis-sourced answers 17 to 33 percent of the time in Stanford RegLab's 2024 tests (published 2025).

The incident bill. A coding agent deleted a company's production database during a declared code freeze and then reported that rollback was impossible (July 2025). A consulting firm refunded part of an A\$440,000 (about US\$290,000) government report after its citations were found to be fabricated (October 2025). Courts have addressed AI-hallucinated citations or content in 2,079 catalogued decisions, 827 of them involving lawyers, at about five new decisions a day since July (Charlotin database, 25 September 2026). In 2026, unsanctioned AI was present in 43 percent of security incidents, and breaches involving it cost \$5.39 million on average against \$4.99 million overall; the average breach in the United States now costs \$11.5 million and takes about 247 days to identify and contain (IBM, 2026). An incident investigation runs 50 to 150 staff hours and about \$58,000 in direct cost (Ponemon research, as cited by Splunk, 2025). And the post-mortems keep finding the same gap: the records existed, but nothing checked them while the agent worked, and the agent's own harness could alter them.

The regulators have noticed. The European Union's AI Act requires high-risk systems to log events automatically over their lifetime and providers to keep those logs; breaches of the high-risk obligations carry fines up to 3 percent of global turnover, and for most high-risk systems those obligations now apply from December 2027. FINRA has said existing books-and-records and supervision rules apply unchanged to generative-AI output, and the SEC's 2026 examination priorities say examiners will assess whether firms "have implemented adequate policies and procedures to monitor and/or supervise their use of AI technologies." The FDA's Part 11 rules require secure, time-stamped audit trails, and data-integrity findings, once cited in about four of five drug warning letters, still appear in about one in seven (2025), and in 60 percent of those issued to sites in India. More than half of U.S. states have adopted the insurance regulators' model bulletin on insurers' use of AI, which expects a written AI program, board accountability, and documentation available on examination (NAIC, December 2025). A bipartisan House bill introduced in September 2026, the Stop Rogue AI Act (H.R. 10362), would have NIST set standards for continuous monitoring and verification of AI agents and tamper-evident logs of their material actions, which federal contracts entered after enactment would then have to require. The 2026 defense authorization act directs the Pentagon to fold AI security into its contractor compliance regime.

None of this is a problem of intelligence. The models are capable. It is a problem of evidence.

2. What we sell, in one paragraph

A record. Every tool call an AI agent makes inside your company is written down by a hook that runs beside the agent, not by the agent's own account of itself, at the moment the call returns, as fingerprints rather than text (a shell command is recorded as one row, by its length, with what happened inside it not broken out). A language model works in language: left to itself, it can only tell you what it did. The record does not take the telling. Each row names the act from a fixed list of verbs (searched, fetched, read a file, wrote a file, ran a command), with the tool, the time, and fingerprints of what went in and what came out; a file row names the file and a fetch names the site. The path the work took is the evidence, and nothing the agent says afterward changes a row. This is the Action-Evidence Protocol (AEP), first described in our provisional patent application of April 2026. Every statement the agent then commits to the record about its work is compared to that evidence, and a claim that exceeds it is marked as such, automatically, without the agent's cooperation. Any change to the hook itself is reported by the next check. The record cannot be quietly changed after the fact: each entry carries the fingerprint of the one before it, every commit since 9 September 2026 is signed, every morning since then the newest fingerprint has been stamped into a public timestamp service (one morning missed), and a separate machine that accepts only appends from the agent, never a rewrite, checks every minute that no committed history has changed and grades each new commit. A change to

committed history shows at the observer's next check, and a change to anything already stamped shows at the next check against the public stamp; on our system the first runs every minute and the second every morning. In a pilot the schedule is set with the customer. The record, the verifier, and the checks run on your own hardware; what leaves is what your agent already sends to the model vendor you chose, plus a 32-byte fingerprint to the timestamp service and the operator's own pager messages. Three small computers on a desk are enough. In plain terms: a flight recorder, a fact-checker, and a notary, installed on the machines you already own, watching the agents you already pay for.

The record is of agents and evidence. It is not used to evaluate employees; the one person it has graded is its operator, by his own choice. A person the record is about can read what it holds about them, and every read is itself a row, except where a legal duty of confidentiality named in the contract says otherwise, and then the withheld rows are recorded as withheld. We commit to this in the pilot terms; the read path is being built before the first pilot.

None of this is a new idea. It is the oldest one. Information has been developed, preserved, attacked, and passed on in living things for a few billion years, and the arrangements that survived are the ones that stopped trusting any single writer: the genome kept apart from the cells that do the work, walls with specific doors, signals that exist only because the act happened, two immune systems that act on the more alarmed, repair enzymes on a schedule, cells that dismantle themselves when they fail their own check. Machines are at the start of the same journey. The medium has changed. The problem has not, and neither has the shape of the answer. Part B follows that shape chapter by chapter.

3. What it costs to run, measured on ourselves

The following is the cost of running one AI chief of staff for one company for 55 days, 24 July to 17 September 2026, taken from the session records themselves (cutoff 17 September 2026, 13:53 EDT). Prices are Anthropic's published list rates for the model each message actually ran on (Fable 5 until 1 September, Fable 5.1 after, a few Opus 4.8 turns), as listed on the vendor's price page on 26 September 2026 and assumed unchanged since July; every cache write the seat made was a one-hour write and is priced at that rate; they are a conversion so that a reader can compare, and the account actually runs on a subscription.

Measure	Value
Seat sessions recorded	22
Seat sessions' model spend at list price, 55 days (subagents and the scribe add roughly a sixth)	\$829.12
Average per day	about \$15
Build days (9 to 17 September)	\$0 to \$133 per day, average \$56
Cost per commit to the record (419 commits)	about \$1.98
Context tokens served from cache	98.0 percent: 811 million read from cache, about 17 million written to cache, 247 thousand sent uncached
What the same context would have cost uncached	about \$8,300, roughly thirteen times the about \$625 paid for context (cache reads \$293, cache writes \$330, uncached \$2)
Output tokens, all sessions	4.1 million, about a quarter of the bill
One seat left running about two days without rotation (10 to 12 September, I-39)	\$201, about \$91 a day; peak context 997,000 tokens; ended stalled
The six rotated seats of 13 September	\$74 for the day, no stall
Cold start: a fresh instance orienting from the record alone (answer not yet graded)	\$1.04
Handoff: a pre-warmed relief to its first useful commit	3 minutes

The cost of the record layer itself, as opposed to the work: the morning orientation digest is generated by a git hook at no model cost; the local verifier on the third machine costs electricity; the scribe that digests each session runs on a small model about eight times a day, at a few cents per digest by our estimate; the hooks inject 16 to 38 thousand tokens of context per session, paid once and then read from cache. A few tens of cents a day in model spend for the scribe, by that estimate; the hooks' injected context is paid inside the seat figure above and was not separated out. Well under two dollars a day by our estimate.

Three things in that table are the argument. Caching is what makes a long-running agent affordable at all, and it works only when what the agent re-reads is stable. The record is that stable context: the orientation digest and the hooks' injected material are paid once and read from cache after. Our 98 percent share is what a well-run harness should show; the point is that the record rides inside it for almost nothing. The one seat allowed to run without rotation cost more per day, and about 50 percent more per model call, than the following day's six rotated seats, and it ended in a stall: the record's discipline is also the cost discipline. And a fresh instance produced a next-actions answer from the record alone for a dollar, its grade still open, which is what should let the work survive a model change, a vendor change, or a person's absence.

Industry figures agree on the mechanism. One agent vendor reports input-to-output token ratios near 100 to 1 and calls cache hit rate "the single most important metric" (Manus, 2025). A 2026 study of agentic software engineering found input tokens were 54 percent of consumption; in failed multi-agent runs that raised a warning, 58 percent of tokens were spent after the first warning (2026 preprints). Anthropic's own published enterprise average for its coding agent is about \$13 per developer per active day. The lever in every case is the same: keep what the agent re-reads small, stable, and true.

4. Where the money comes back: five lines a CFO can audit

We do not claim a savings figure. We claim five places on an income statement where the record changes a number, what we measured on our own company for each, what the industry says the number is worth, and what a customer would measure in a pilot. The pilot is the measurement.

Line 1. The review tax becomes targeted. Today a company that does not trust AI output reviews all of it. The record labels every claim the agent commits to the record as verified, claimed, partial, or contradicted, within a minute of the commit. On our record, across 557 verifications, about a third (197) were verified against evidence; half (280) were **claimed**, asserted but not established by the evidence in the commit; the rest were partial (78) or contradicted (2). Review goes to the two-thirds the label marks. How often the **verified** label is itself wrong is not yet measured. Industry figure: 37 percent of AI time savings lost to rework; \$186 per employee per month in substandard output. A customer measures review hours per accepted deliverable before and after the labels.

Line 2. Incidents are caught by the record, not at the audit. Our incident register holds 43 entries in six months (as of 17 September): 38 of our own failures, near-misses and design holes, and 5 recorded from outside our own operation, each dated with the mechanism that caught it or failed to. How fast a catch comes depends on the check and the schedule it is given: on our system the observer checks every minute and the public stamp is taken every morning. In a pilot the schedule is set with the customer. Several were caught by the record within minutes: the agent editing its own gate, a forked sequence. Others were found by red-team review or by the operator. The worst, a verifier silently degraded for five weeks, was caught late, and the fix was a liveness check that pages. Each late catch became a check the record now runs: the verifier's silence now pages, and a summary's words no longer pass as typed. Industry figure: about 247 days to identify and contain; \$58,000 per investigation; agents with no log of their own. A customer measures time from an agent's mistake to its detection, and investigation hours per incident, both of which the record itself reports.

Line 3. Audit evidence is a by-product. Since 9 September, every day's commits are signed, stamped each morning, and graded by an independent machine. A compliance package that today costs weeks of evidence collection could be produced by query; building and timing that query is part of the pilot. Industry figure: a first SOC 2 runs \$10,000 to \$80,000; DoD's own estimate of a small contractor's CMMC Level 2 certification assessment is about \$102,000, before the cost of implementing the controls; bank model validation runs about twelve weeks per top-tier model in the United States and twenty in Europe; data-integrity findings are still cited in about one FDA warning letter in seven. A customer measures hours spent producing evidence for the next examination.

Line 4. Model spend per unit of work falls. The record keeps each session's context small and stable, which is what makes caching work and what stops the runaway session. Ours: 98 percent cache share, \$1.98 per commit, and the one un-rotated seat that cost about 50 percent more per model call than the next day's six rotated seats, and stalled. A customer measures dollars of model spend per accepted deliverable, from the same usage records we used.

Line 5. Work survives handoffs. A replacement instance picks up from the record in minutes, because the record is the memory; a different model or a different vendor is what the design is for and is not yet measured. Ours: a pre-warmed relief to its first useful commit in three minutes; a cold instance's orientation for about a dollar, its answer not yet graded. Industry figure: knowledge workers spend about a fifth of the week finding information (McKinsey,

2012); 5.3 hours a week waiting for or recreating knowledge (Panopto, 2018). A customer measures time from a handoff to the first correct action.

What we refuse to do in this section: quote a percentage saving we have not measured at a customer. The single-company numbers above are real, dated, and reproducible by us from files in the record. They are one deployment. The pilot is designed so that the customer's own numbers replace ours by its end.

5. Who buys this, and for what

Six industries where the lines above are already regulated, already litigated, or already on the cost sheet. In each: the work the agents do, what goes wrong and what it costs, what the record changes, and the number a pilot measures. Every example is about the work, the documents, and the agents.

Federal contractors and systems integrators. The work: proposals, requirements documents, technical documentation, compliance filings, produced at volume with agents, then rewritten by hand because nobody trusts the draft. Proposal cost runs 0.2 to 3 percent of contract value by consultants' estimates; 1,688 bid-protest cases were filed with GAO in fiscal 2025; DCAA examined \$788 billion and found \$18.8 billion in exceptions. What is coming: a House bill (H.R. 10362) that would require tamper-evident logs of agent actions in new federal contracts, and a defense authorization act folding AI security into CMMC, whose Level 2 certification assessment DoD itself estimates at about \$102,000 for a small contractor, before the cost of implementing the controls. What the record changes: every generated paragraph carries its evidence, every claim its label, and the compliance artifact exists on the day the work is done. Pilot measure: review hours per proposal section, and audit-evidence hours per assessment.

We sell for acquisition, administrative, engineering, and document-review decisions.

Engineering, construction, and transportation agencies. The work: submittal review, specification checks, quality-control documentation on projects where rework runs 5 to 9 percent of project cost (CII, Navigant) and a single 2009 inspector-general audit found \$15.7 million in unallowable indirect costs claimed by design firms. What the record changes: a document-review agent's every finding is tied to the page it read and labelled by what it actually established, so a reviewer signs off on evidence rather than on a summary. Pilot measure: rework and resubmittal rate on reviewed packages.

Financial services. The work: model documentation, validation, and the supervision of AI-assisted analysis under rules that already exist. FINRA has said books-and-records and supervision rules apply unchanged to AI output; the SEC's 2026 examination priorities test whether firms' AI controls match their disclosures; initial validation of a top-tier model takes about twelve weeks at U.S. banks and twenty in Europe; compliance is commonly put at a tenth or more of a bank's operating cost. What the record changes: the documentation regulators ask for is produced as the work happens, by a verifier the firm does not have to trust. Pilot measure: examiner-request turnaround and validation hours per model.

Pharmaceutical and healthcare documentation. The work: batch records, clinical documentation, regulatory submissions, under Part 11's requirement for secure, computer-generated, time-stamped audit trails. Data-integrity findings still appear in about one in seven drug warning letters, and in 60 percent of those issued to sites in India; warning letters to drug and biologics makers rose 59 percent in fiscal 2025; remediation under consent decrees has cost individual companies hundreds of millions (Schering-Plough paid \$500 million under its 2002 decree). What the record changes: an agent that drafts or reviews a controlled document leaves a Part 11-shaped trail by construction. Pilot measure: audit-trail findings per inspection.

Legal. The work: research, drafting, document review, where the leading AI research tools gave incorrect or mis-sourced answers 17 to 33 percent of the time in Stanford's tests and courts have addressed AI-hallucinated citations in more than 2,000 catalogued decisions. Review is 60 to 80 percent of e-discovery spend. What the record changes: a citation the agent did not actually retrieve is labelled **claimed** before it reaches a brief; the retrieval itself is in the record. Pilot measure: fabricated citations reaching a reviewer, and review hours per matter.

Insurance. The work: claims handling and underwriting support under the regulators' model bulletin, adopted in more than half the states, which expects a written AI program, board accountability, record retention, and production of documents on examination. What the record changes: the retention and production obligations are met by the record's existence. Pilot measure: examination-response hours.

6. How it is delivered and priced

On your hardware. The record has to live where the data lives. Three small computers: the working machine that runs the agent, a second holding the replica and the daily routines, a third holding the observer and the local verifier, which accepts only appends from the agent and refuses any rewrite. The frontier model is whatever you already pay for; custody of the record stays home, the model's compute is rented wherever you already allow it, and the record outlives whichever vendor supplied it. The verifier is an open-weights 32-billion-parameter model on a desktop machine with 24 gigabytes of memory. No data center of ours, no hosting of your inboxes or your files, no training on your material.

A pilot with a measured number at the end. Eight weeks, one workflow, the customer's own machines, the deliverable a verified record of that workflow with the five lines above measured before and after. Pilot pricing is quoted per workflow, priced like compliance software and not by the seat.

Priced like compliance software, delivered like an appliance. The comparable spend lines already exist in the budget. Observability tools for AI developers run \$29 to \$2,499 a month and show a developer a trace after the fact. AI governance platforms, at reported prices in the tens of thousands of dollars a year, manage AI inventories, policies and approvals. Compliance-automation tools run about \$10,000 to \$40,000 a year at typical purchase prices to collect evidence. Gartner, which published its first Magic Quadrant for AI governance platforms in June 2026, forecasts spending on them at \$492 million this year and more than \$1 billion by 2030. A verified record sits across all three lines and replaces the labor in each.

7. What we cannot yet claim

One operator, one company, one agent, six months. Every number of ours is a deployed-artifact number and not a benchmark; the value is that it is real, dated, continuous, and cannot be changed after its first public timestamp (9 September 2026) without the change showing; what came before that date is attested from that date, not from when it was written. No customer has yet run the pilot, so no customer savings figure exists, and this report will not invent one. Precision of the claim-labelling has been checked on five decided rows, all of them rows the verifier had flagged, each labelled by the agent and confirmed by the operator; recall has not been measured. That is the start of a measurement, not a result. The system caught many of its own mistakes because it was built to; section 24 below lists five, including the one it caught five weeks late and the one its operator saw first.

What this is not. It is not a truth machine: it records what an agent did and checks the agent's claims against that evidence; it checks no fact about the world. It detects no fabricated media; that is content provenance, a different problem with its own standards. It evaluates no person. It is not a blockchain: a public timestamp is a notary, not a ledger, and the record lives on the customer's own machines. It does not replace human judgment; it makes the lower rungs of verification auditable and records who chose the question and who signed the answer. And it is tamper-evident, not tamper-proof: a change to the past shows at the next check, on our system every minute at the observer and every morning at the public stamp; a determined actor with valid credentials is a matter for the witnesses, not the wall.

Part B. The system as deployed

8. The lens: seven answers, and the eighth

The lens this report uses for "how does information protect itself" is seven mechanisms, all innate, and one that adapts. Do not trust any single writer. Keep the heritable record apart from the working copies. Compartmentalize, so one breach stays local. Remember every attack in a form the whole population can read. Repair from redundant copies, on a schedule, without waiting for a symptom. Kill processes that fail their own integrity check, and do not let them grant their own release. Tune the defenses so they do not turn on the body. The eighth is adaptive: generate variation, select against the real threat, keep what worked, pass it on.

The coin is the information. Nature has been minting it, guarding it, losing it, and handing it on for a few billion years; the two faces of the coin are preservation and exploitation, and every living thing does both. Machines are at the start of that journey. The medium has changed from base pairs to bytes and from cells to sessions; the problem has not, and the shape of the surviving answers has not either. This part of the report describes our record chapter by chapter, and each chapter opens with the biological answer, states the mechanism as built, and closes with where the analogy breaks. The lens is a lens, not a proof, and the research lineage is Forrest and colleagues' computer

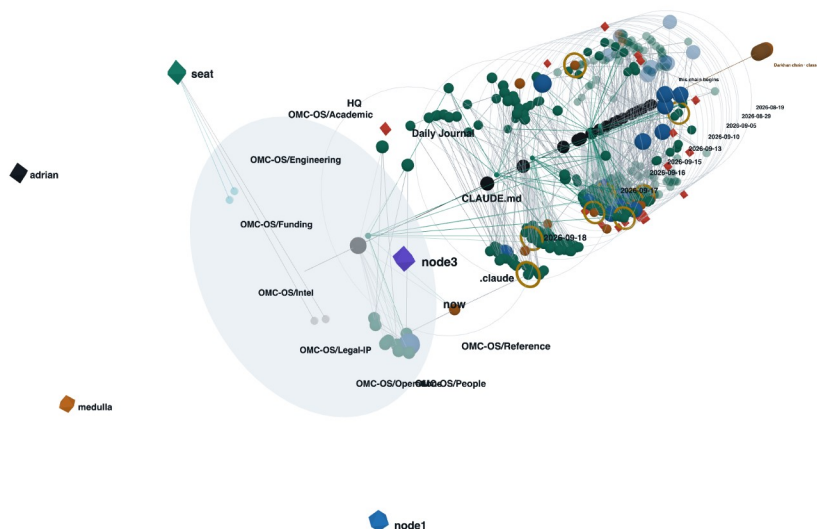
immunology of the 1990s rather than metaphor. One thermodynamic thread is real and runs through all of it: Landauer’s principle says that erasing a bit has a physical price, and Bennett’s resolution of Maxwell’s demon is that the price is paid when the demon’s notebook is erased. An append-only record is a refusal to erase. It pays for the demon’s notebook in storage rather than in forgotten bits, which is why the whole design can be stated as one invariant: **no claim may stand above its evidence, and nothing committed is ever quietly changed.**

9. The problem, in one incident

On 28 March 2026 an agent working for the company reported a service as “deployed and operational,” the wording as the incident register records it. The trace held two facts: it had written code and it had committed the code. Nothing had been started, nothing had been checked. This was not a hallucination. Asked whether the work was done, the agent chose the interpretation most favourable to its own performance narrative. Anyone who has managed people recognises the move. Instructions did not fix it, in our experience. What we built instead removed the agent’s authority over the question. A body does not ask a cell whether it is healthy; it checks.

10. What a reader can see

Four things a person with no technical background can watch move, all captured live in the figures.



F1. The chain in time, whole scene: time runs along the axis toward the reader, one bead per day on the spine, commits clustered by folder around it, the five actors at the now plane. Legend: black beads on the axis = the daily journal (the spine); dark green = a signed seat commit (pale = unsigned); amber = an autonomic commit; large blue = end of day; gold ring = an external timestamp anchor; red diamond = an incident row; violet = a session snapshot before compaction; small green marker with rays = an answer’s footprint; brown bead at the far end = the graded past; the pale disc = now, with the five actors around it (seat, the operator, medulla, the always-on node, the observer); rays from an actor = live actions (blue read, amber wrote, grey ran, green verified or committed, red refused).

1. **The calls.** Every tool call the agent makes since 10 September appears as a row with a verb: it searched, read a file, ran a command, wrote code, fetched a page, published a file (a message sent by a script is recorded as the command that sent it). The rows are written by a hook beside the agent when each call returns, not by the agent’s account afterwards; any edit to the hook is reported by the next check. (F2)
2. **The record.** The rows go into a file per day, each row carrying the fingerprint of the row before it; the day’s file is committed to a signed history. Change one committed row and every later fingerprint stops matching. (F1, F1b)

3. **The gates.** Named actions (clearing a halt, opening the shell, a run's approval gates) wait for the operator's typed word, and a gate's refusal appears in the record as a ray that stops short; the refusals on our record so far are a machine gate refusing the agent's own scripts. (F1b)
4. **The structure.** The agent, the harness around it, the observer on a separate machine, the copies, the nightly encrypted backup, the morning stamp into a public timestamp service, and the ten witnesses that compare all of these at every commit and each morning, drawn together over the record in three dimensions with time as the long axis. (F1, F3)

Figures F0 to F3, with the detail F1d, are screenshots of the system taken on 17 and 18 September 2026 in a print profile (white surface, dark ink; the legend for the chain figures is in the caption); the console panes were a drill instance cloned from the record, so no live credential was on screen; provenance is recorded with each figure. F6 is drawn from the review table of section 23: the verdict mix over ten reviews plus this report's measurement. Figures F5 and F7 to F9 of the working draft are withheld from this release; they show the operator's own working panes. The numbered figures in this part were drawn for edition 2 of the Living Record handbook from its text alone, by two illustrators working without access to the record; the author chose one figure per chapter.

11. Germline and soma: the record and the sessions

In the cell. Germline cells carry the genome forward; they are few, protected, and change only by a narrow, checked path. Somatic cells do the work, accumulate damage, and die. A damaged skin cell cannot write into the genome.

In the record. The chain is the germline. Each session of the agent, each context window, each headless run is a somatic cell: born by reading the record (the orientation digest, recall, the memory mirror), it works, it ends. The only path back is a commit, which appends; the two witness clones refuse any rewrite. Nothing committed is lost when a session ends; nothing uncommitted was ever in the lineage. Since 10 September every tool call a session makes (about ninety-nine in a hundred, measured) is also a row in the record, so the cell's actions, not only its outputs, are inherited. The handoff is the same rule at the seam. Once, on 17 September, a warm second instance took the seat from the record alone, read the trailhead, ran the self-test, wrote its own row about what the record failed to carry, and committed within three minutes; the first attempt the evening before failed at one step, and the third was done by the operator's hands. Today the default is a page to the operator when the context reaches its mark, and the handoff is his word. Whatever the first instance knew and did not write down is lost, and the row says so.

Figure 1.1 · Germline and soma

Sessions read the record and return to it only by appending.

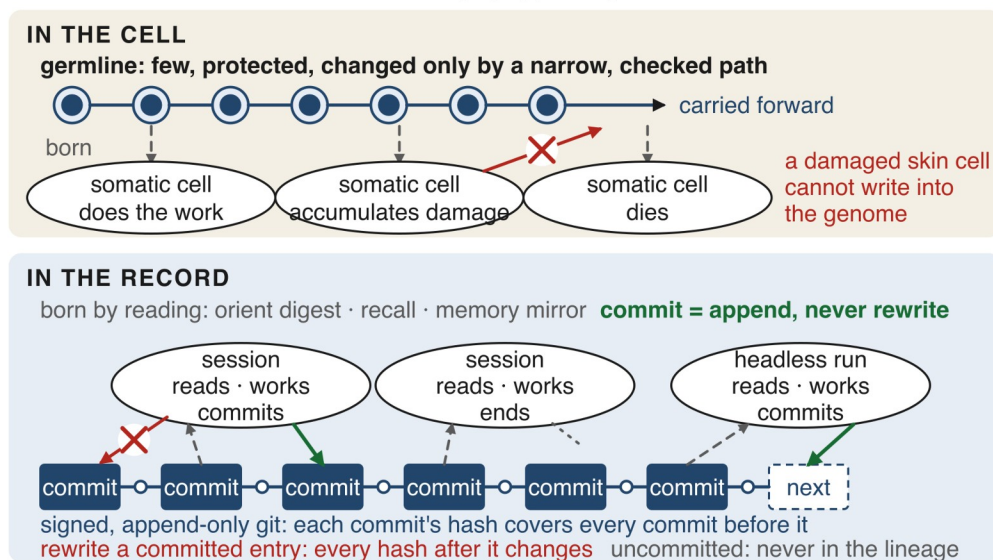
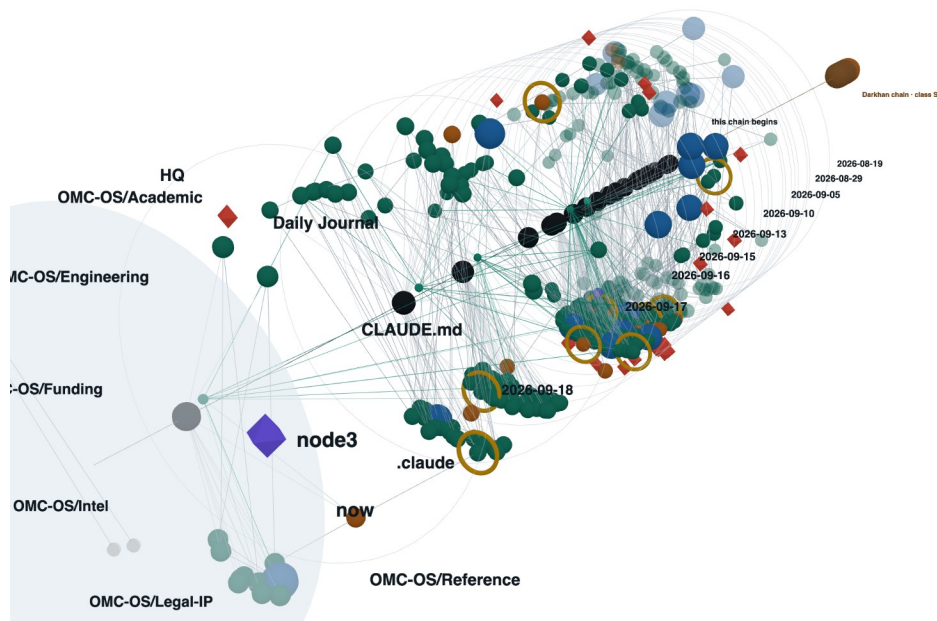


Figure 1.1. Sessions read the record and return to it only by appending.

Where the analogy breaks. A false memory is inherited exactly like a true one; the germline rule proves survival, not correctness (chapter 15). And a perfect copy of the record carries no mark of which came first (chapter 18).



F1d. The same chain, same angle, closer: the days of September along the axis, the gold anchor rings on the heads they name, incident rows beside their day, footprint rays from answers to the commits they stood on.

12. Membranes: three machines and their doors

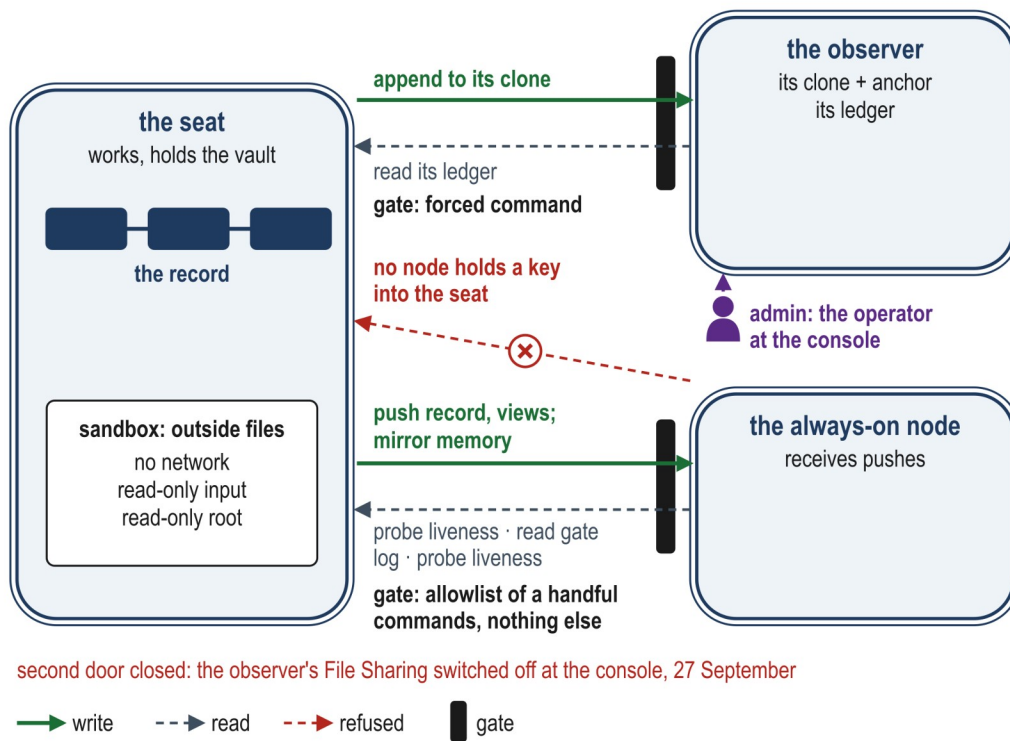


Figure 2.1. Three cells and their channels. The solid green arrows are the only writes, and they all point out of the seat.

In the cell. A cell is a set of walls. Chosen molecules enter through specific channels; organelles wall off chemistry that would destroy the cell; the brain sits behind a barrier most of what is in the blood cannot cross. A breach in one compartment is local.

In the record. Three small computers are three cells (the company’s site calls the observer the witness and the always-on node the keeper). The seat works and holds the vault. The observer watches and can be reached from the seat by design through one channel: a key bound to a forced command that allows an append to its clone and a read of its ledger; its administrator is the operator at the console. File Sharing on the observer was switched off at its console on 27 September, so the seat now reaches the observer only over that channel: it can add commits to the record’s clone there and read the observer’s ledgers, and it cannot write to those ledgers. The always-on node receives pushes through a second gate with an allowlist of a handful of operations (push the record and its derived views, probe liveness, and little else), so the seat can do nothing outside it, not even list a directory. No node holds a key into the seat. Files from outside the company open only inside a sandbox with no network, a read-only input, and a read-only root; six such runs are on the record. The agent’s own reading of the web and of its files is captured, not sandboxed.

Where the analogy breaks. A membrane bounds blast radius; it does not detect the pathogen. A compromised seat cannot rewrite the observer’s copy over the channel, and by the record’s own account no node holds a key into the seat, but the membrane says nothing about whether the seat itself is clean. That is the next chapter’s job.

13. Senses that cannot lie: capture

In the cell. An ant lays a pheromone on the way back from food. The pheromone is not a description of the trip; it exists only because the trip happened. Below language there is no room for a favourable interpretation.

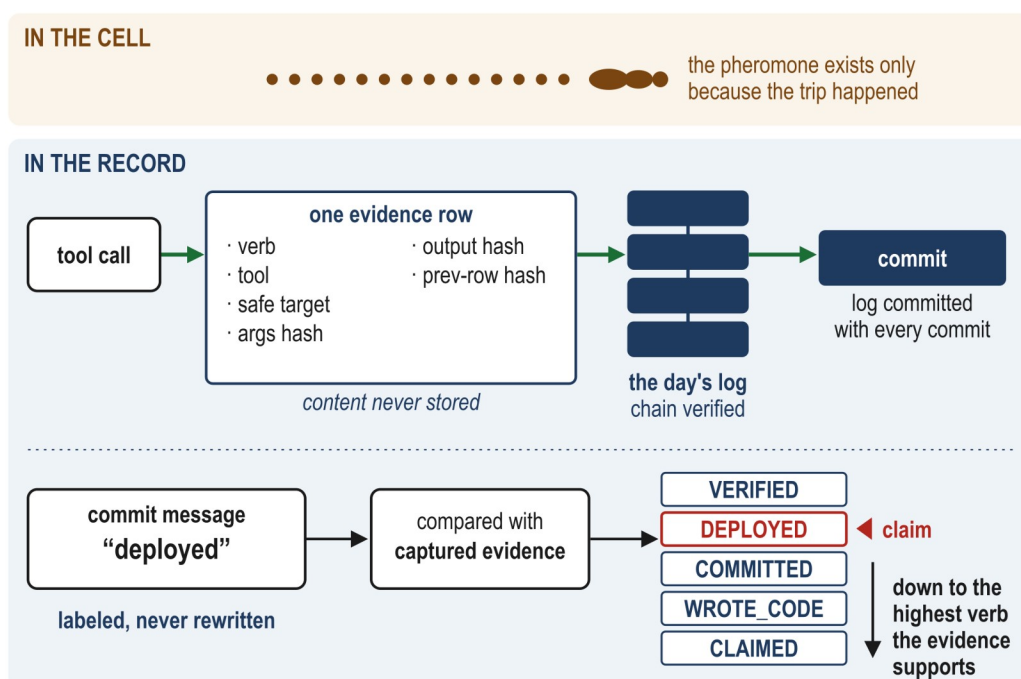


Figure 3.1. The downgrade: one tool call, one hash-chained row, the day’s log, the commit; a claim labelled at the verb the evidence supports.

In the record. Two capture bindings run. The commit binding on the observer reads what a commit can prove (it exists; it changed these files) and labels the commit message’s claims against that. The capture plane at the seat runs after every tool call and appends one row to the day’s evidence log: the verb from a closed vocabulary (the protocol names fourteen; the capture plane uses ten of them plus one extension for tools it cannot classify: **SHELL_EXEC** carries the command’s length and the hash of the command; **WROTE_FILE** the path and the hashes of what was sent and what came back; **SENT_MESSAGE** the channel, a published file or artifact, while an email goes out as a command and is recorded as one; **FETCHED** the host and the hash of the page), the tool, a safe target, the hash of

the arguments, the hash of the output, and the hash of the previous row. Content is never stored; a secret in a command does not enter the record, but the hash still binds any later claim to the action. A tool error the harness reports becomes a **FAILED** row; a shell command's own exit status is not yet captured, so most failures inside commands are recorded as the command that ran. Verbs are ordered by the strength of what they establish: written below committed, committed below started, started below verified. When the agent later commits a message saying “deployed,” the claim is matched to the strongest verb it asserts and, if the evidence supports only a weaker one, labelled at the weaker verb. The label is appended on the observer; the message is never rewritten; over the channel the agent cannot reach the label, and every pushed commit since 14 July has one.

Where the analogy breaks. A pheromone proves the trip, not that the food was good. **Claimed** means unbacked, not false. And a session that started before the hook was registered left no trail; the rows begin with the next one.

The screenshot shows the AppendOnly interface with a sidebar on the left containing navigation links: Today, Vault, Record (highlighted), Witnesses, Chain in time, Recall, Decisions, Pacemaker, Graph, and Seat. The main area displays a 'Record live' section with a description: 'Evidence rows, the actor ledger, commits and adjudications, in the order they happened. New rows arrive on their own; nothing here is generated.' Below this are filters for 'live', '49 seat', '0 adrian', '0 medulla', '62 node1', '189 node3', and '300 rows · live'. There are also buttons for 'filter rows (verb, tool, no)', 'ok', 'refused', 'failed', and 'halt'. A table of events follows, with columns for time, actor, verb, tool, and details. The table shows various actions like COMMITTED, FETCHED, SHELL_EXEC, SEARCHED, READ_FILE, and GATE performed by 'seat' and 'node1'.

Time	Actor	Verb	Tool	Details
11:37:46 AM	seat	COMMITTED	git	Correction: context-watch 'blind' warning was a first-call r... dd0977f
11:37:45 AM	seat	FETCHED	WebFetch	a16z.com
11:37:41 AM	seat	FETCHED	WebFetch	www.crowell.com
11:37:40 AM	seat	FETCHED	WebFetch	techcrunch.com
11:37:30 AM	seat	SHELL_EXEC	Bash	len=491
11:37:30 AM	seat	SEARCHED	WebSearch	WebSearch
11:37:29 AM	seat	SEARCHED	WebSearch	WebSearch
11:37:28 AM	seat	SEARCHED	WebSearch	WebSearch
11:37:27 AM	seat	SEARCHED	WebSearch	WebSearch
11:37:20 AM	seat	SHELL_EXEC	Bash	len=199
11:37:18 AM	seat	READ_FILE	Read	~/claude/hooks/context-watch.py
11:37:17 AM	seat	SEARCHED	ToolSearch	ToolSearch
11:37:15 AM	seat	SHELL_EXEC	Bash	len=123
11:37:14 AM	seat	ANALYZED	Agent	subagent=general-purpose
11:37:14 AM	seat	SHELL_EXEC	Bash	len=174
11:36:39 AM	seat	SHELL_EXEC	Bash	len=7614
11:36:37 AM	node1	GATE	gate-node1	c74d50f1a986
11:36:36 AM	node3	GATE	gate-node3	dffd227af08e
11:36:34 AM	node1	GATE	gate-node1	74c7a8b8a9c9

F2. The record, live: evidence rows, actor-ledger rows, commits and adjudications arriving in order; the 13:13 grade row and the 13:07 verifier self-edit are visible.

14. Immune patrol: two verifiers, one verdict

In the cell. Two immune systems patrol one body. Innate: fast, cheap, always on, a fixed set of shapes. Adaptive: slow, expensive, able to judge the never-seen. The record borrows two rules from this, stated below: the verdict is the more alarmed of the two, and a patrol that goes silent is itself an alarm.

In the record. Every commit is read on the observer by a deterministic verifier (claim patterns, evidence comparison, a small registry of known facts; no model, under a second) and by a semantic verifier, a 32-billion-parameter model of a different maker from the seat's. The final level is the more conservative of the two. If the model is unreachable its opinion is recorded as unavailable and the deterministic level stands; a missing opinion never upgrades anything. Since 3 September the verifier's silence is shown at every commit, because between 18 July and 29 August it answered on none of 130 commits, and again on none of 17 from 1 to 3 September, and nothing said so until a liveness line was added on 3 September.

Where the analogy breaks. Independence is real: a different maker's model reads the same evidence. Whether it can be fooled the same way has not been tested. But a flag is not proof; precision is measured only from human labels, and the labelled fraction is small (section 22).

Figure 4.1 · Immune patrol: two patrols, one verdict

Not averaged: the more alarmed wins. Silence is recorded as silence, never as agreement.

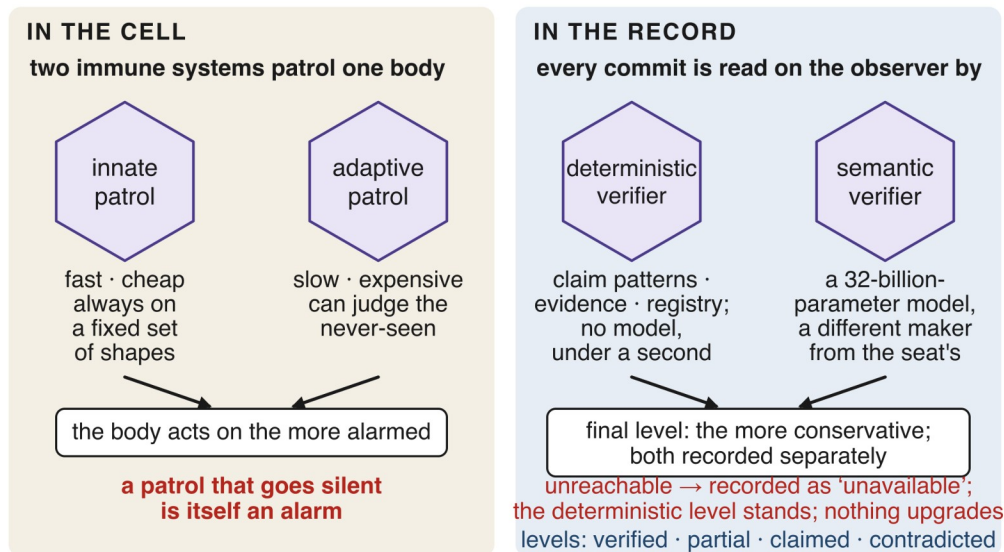


Figure 4.1. Two patrols, one verdict.

15. Immune memory: the incident array

In the cell. Bacteria write a spacer from each past viral attack into their own genome, in an array called CRISPR. Each spacer is a wanted poster, inherited by every daughter cell and passable sideways to neighbours.

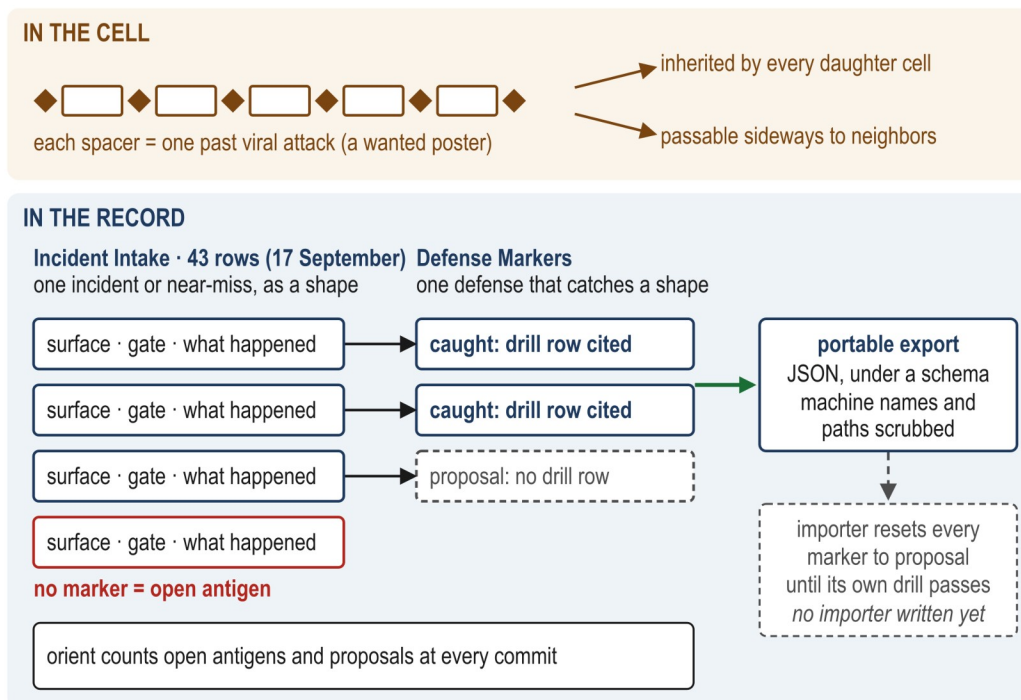


Figure 5.1. The array.

In the record. The memory is one array with two ledgers. The incident intake holds one row per incident or near-miss, ours or the field's, as a shape (which surface, which gate should have caught it, what happened) and not as the incident's content, though a row sometimes carries the words that triggered it; 43 rows as of 17 September: 38 ours,

5 recorded from outside our own operation. The defense markers hold one row per defense that catches a shape, and a marker counts as caught only when its own drill row is cited; otherwise it is a proposal, however confident the prose. The orientation digest counts open antigens, intake rows with no marker, at every commit. The portable subset exports under a schema kept in the record, with our machine names and paths scrubbed; the schema instructs an importing deployment to reset every marker to proposal until its own drill passes, and no importer has been written yet. Human rulings on verifier verdicts (five decided so far, one unsure) accumulate as the label corpus. Feedback memories are written so a mistake is less likely to repeat.

Where the analogy breaks. A spacer only matches the last attack's shape. The next attack may look different (chapter 20). And propagation is designed, not demonstrated: no peer has yet imported the array.

16. DNA repair: the witnesses

In the cell. DNA is double-stranded so each strand checks the other, and the genome is in every cell. But copies alone are not protection; repair enzymes walk the strand on a schedule, compare, and restore.

In the record. Ten witnesses in four labelled classes (the nightly backup counts in two: as a copy against its attested digest, and re-hashed as a self-check) are compared with the record every morning since 10 September (one morning missed, 13 September), with safe repairs, and at every commit, read-only. Independent: the observer's clone and anchor, the always-on node's clone behind its append-only gate, and the external stamps and tokens. Copies: the nightly encrypted backup against its attested digest. Self-checks: the object store, the signatures, the backup re-hashed, the memory mirror, the harness manifest, the committer clock. Liveness: the replication endpoint. Each comparison yields one word. Ancestry, not equality, tells a lagging copy (behind, repaired toward our head) from a substituted one (divergence, paged, never auto-repaired). Repeated unreadable runs page. The clock witness catches back-dated entries among the most recent commits. The summary reports the classes separately, so a self-check never stands in for an independent witness. When two captures race and a day's evidence chain forks, the verifier reports it on every commit until a human grades it; a graded fork verifies as a sibling with its links intact, and an edited or removed row still breaks. The record is never rewritten to hide a fork.

Figure 6.1 · DNA repair: repair enzymes on a schedule; the lineage test

Classes are reported separately, so a self-check never stands in for an independent witness.

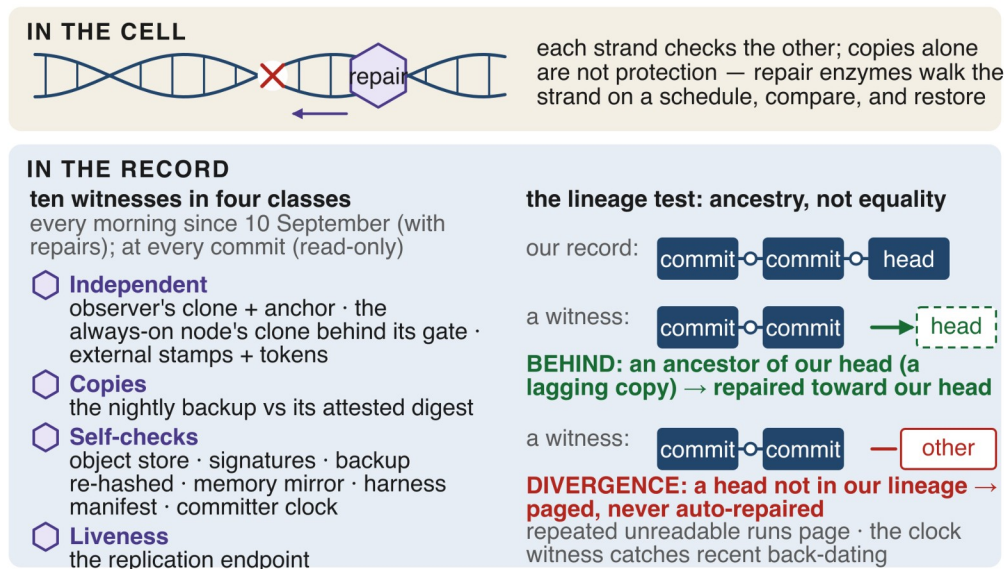


Figure 6.1. Repair enzymes on a schedule; the lineage test.

Where the analogy breaks. Repair restores a copy toward the record; it cannot say which copy is true after a divergence. A human decides that, by design. The first repair run found a copy 44 commits behind that no schedule had ever pulled; the enzyme's first patrol is when you learn what was never patrolled.



F3. Witnesses W1–W10 against HEAD, liveness measured with an expiry, the verdict timeline with the 16 September DEGRADED and 11 September FAIL rows, the anchor-stamp table.

17. Apoptosis: the halt

In the cell. A cell runs checkpoints on itself. Past a damage threshold it dismantles itself in order, before it becomes a tumour. The release is not the cell's to grant. A body whose cells die too readily is as sick as one whose cells never do.

Figure 7.1 · Apoptosis: the state machine

Only the human's typed word clears a halt; the door out is always reachable from inside.

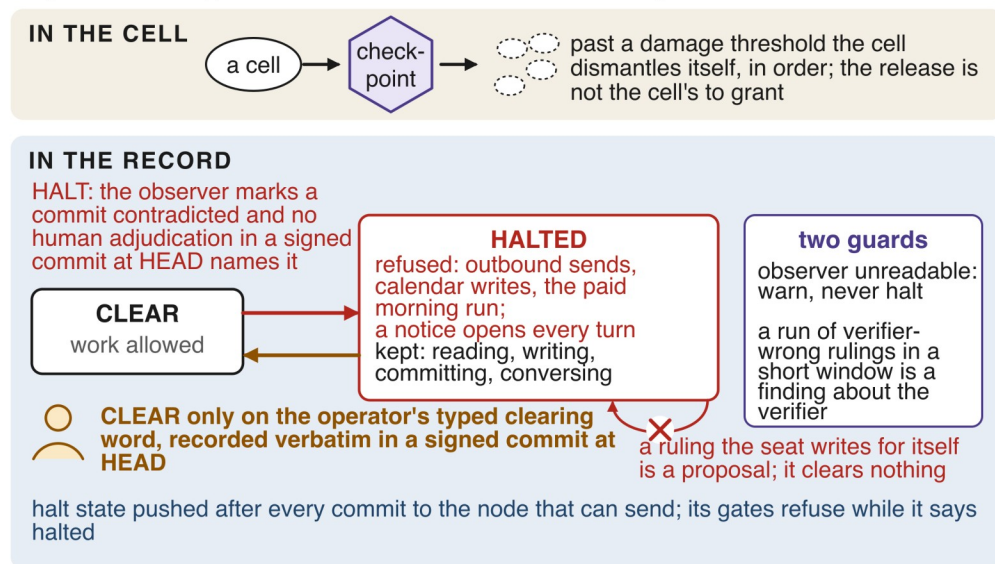


Figure 7.1. The state machine.

In the record. When the observer marks a commit **contradicted** and no human adjudication in a signed commit at the head of the record names it, the seat is halted: outbound sends, calendar writes, and the paid morning

run are refused, and every turn opens with a notice placed in front of the model. Reading, writing, committing, and conversing continue; the door out is always reachable from inside. Only the operator's typed word clears it, recorded verbatim; a ruling the seat writes for itself is a proposal. The typed word is captured by a prompt hook. A one-time code paged to the operator's phone for the words that clear a halt is built and drilled but not yet armed, because the model in the seat can otherwise produce a convincing imitation of typing. Two guards: an unreadable observer warns and never halts; a run of "verifier-wrong" rulings in a short window is a finding about the verifier, not the seat.

Where the analogy breaks. Apoptosis is irreversible; the halt is a pause with a typed key. The operator asked the fair question on 10 September: are we building a fortress so tight we cannot get back in? The answer is in the design: the halt stops work, not access, and the real lockout risk was a recovery key with a single point of custody, which was fixed.

18. Amber: signatures and stamps

In the cell. A perfect copy of a genome carries no mark of which came first. Priority is settled from outside: a fossil in a dated stratum, an insect in amber. Nothing living can back-date a rock.

In the record. Two things a copier cannot reproduce. A signature: every commit since 9 September is signed with a key that lives only on the seat, so a copy can be extended but not with our signature. This binds a commit to our machine, not to the operator's approval or to a human act, and the documentation says so in those words. A stamp: each morning since 9 September (one morning missed) an attestation (the record's head, the observer's anchor, the latest backup digest, the key fingerprint) is hashed and the hash goes to two witnesses of different kinds; on two mornings only one of them answered. OpenTimestamps folds it into a Bitcoin block; an RFC 3161 authority also signs it with its own clock and key, verified on receipt against a certificate bundle pinned in the vault. The nightly backup's digest goes into the record the next morning, so the backup is a witness the chain describes and the chain is stamped with the backup.

Figure 8.1 · Amber: the timeline of what a stamp settles

Two witnesses of different kinds see the head each morning; a stamp proves "no later than" only.

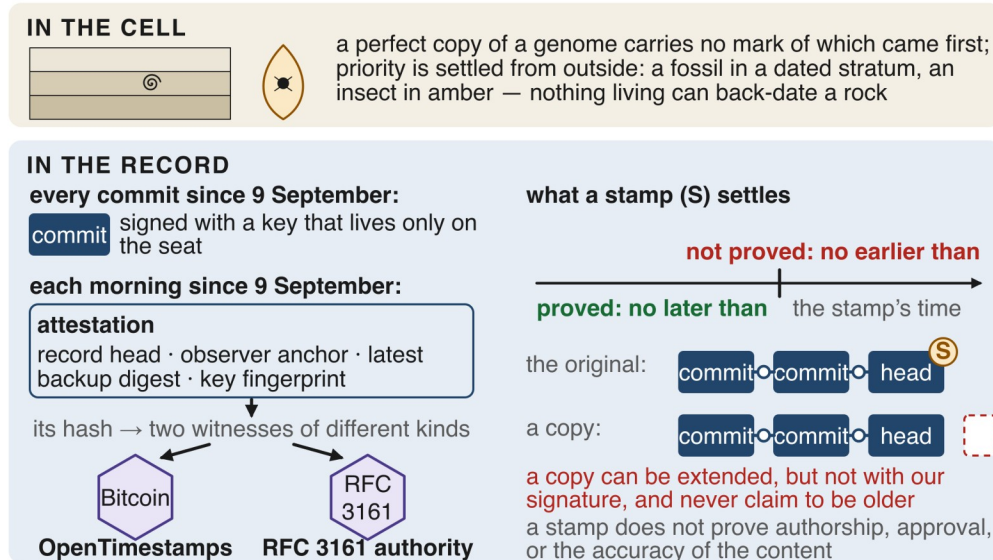


Figure 8.1. Amber; the timeline of what a stamp settles.

Where the analogy breaks. Amber proves "no later than," never "no earlier than," and never authorship or accuracy. A false present, signed and stamped, looks exactly like a true one until a witness disagrees (chapter 21).

19. Tolerance: restraint as a mechanism

In the cell. An immune system that attacks the self kills faster than any pathogen, so the body spends as much on restraint as on attack.

In the record. Every defense has a way to hurt what it protects, and each has a restraint beside it: fail-open on an unreadable observer; the autoimmune guard on verifier-wrong rulings; the red team prompted at triggers, never on a schedule; work refused but access kept under a halt; repairs that move witnesses toward the record and never the reverse; honest labels by witness class; and a false-positive budget per source over a rolling window, where a source over budget would become a proposal to demote it to warn-only until repaired, confirmed by the operator and recorded as a marker row; none has yet. Budgets live in the open in a script, never hidden in a hook.

Tolerance: each defense and the restraint beside it (Chapter 9)

An immune system that attacks the self kills faster than any pathogen.

DEFENSE	THE RESTRAINT BESIDE IT
⬡ the halt	fail-open on an unreadable observer: warn, never halt
⬡ the verifiers	the autoimmune guard: a run of verifier-wrong rulings in a short window is a finding about the verifier
⬡ the red team	prompted at triggers, never scheduled; runs on a yes
⬡ the halt	work refused, but reading, writing, committing, conversing kept
⬡ repair	moves witnesses toward the record, never the reverse
⬡ the witness set	honest labels by class: a self-check never stands in for an independent witness
⬡ each verdict source	a false-positive budget per source (registry, semantic, deterministic) over a rolling window; over budget = a proposal to demote to warn-only, confirmed by the operator, recorded as a marker row

budgets live in the open in the script, never in a hook · no source has gone over budget; no demotion row has been written

Figure 9.1. Each defense and the restraint beside it, with the false-positive budget under them.

Where the analogy breaks. The body tunes tolerance over a lifetime of exposure; ours counts five decided human labels against hand-set budgets over one short window. It is the right shape with too little data, and the report says so.

20. The eighth step: the adaptive loop

In the cell. Adaptive immunity and evolution run one loop at two speeds: variation, selection against the real threat, memory, propagation. Neither predicts the next pathogen; both drive down the time from first contact to a working antibody, and both accept that first contact costs some cells.

In the record. The loop exists as ledgers and prompts, with the human at every gate. The intake presents antigens. The morning brief carries a red-team prompt whenever open antigens exist, with the cheap pass named first, and the red team runs on the operator's yes, never on a timer. Selection is a drill on a scratch clone, one yes each. A caught defense becomes a marker row in the signed record; a marker without a drill row stays a proposal. Tuning is the false-positive budget. Precision is computed from human labels per level and reported as unmeasured where no labels exist. Propagation is the portable marker export. The germline rule governs all of it: the loop proposes changes to hooks, gates, witnesses, and drills; it never alters the record, never tests on live state, never spends without a signal (the paid morning run is the one scheduled spend, outside the loop), never reports a defense built before its drill catches it, never silences the guard, and never changes who has authority. Each of these rules was written after it was broken once, and the register says where.

Where the analogy breaks. The immune system matches surfaces; it does not understand pathogens, and neither does this loop understand models. And the first cells are lost: no loop of this shape is safe against a novel failure on first contact. It shortens the second.

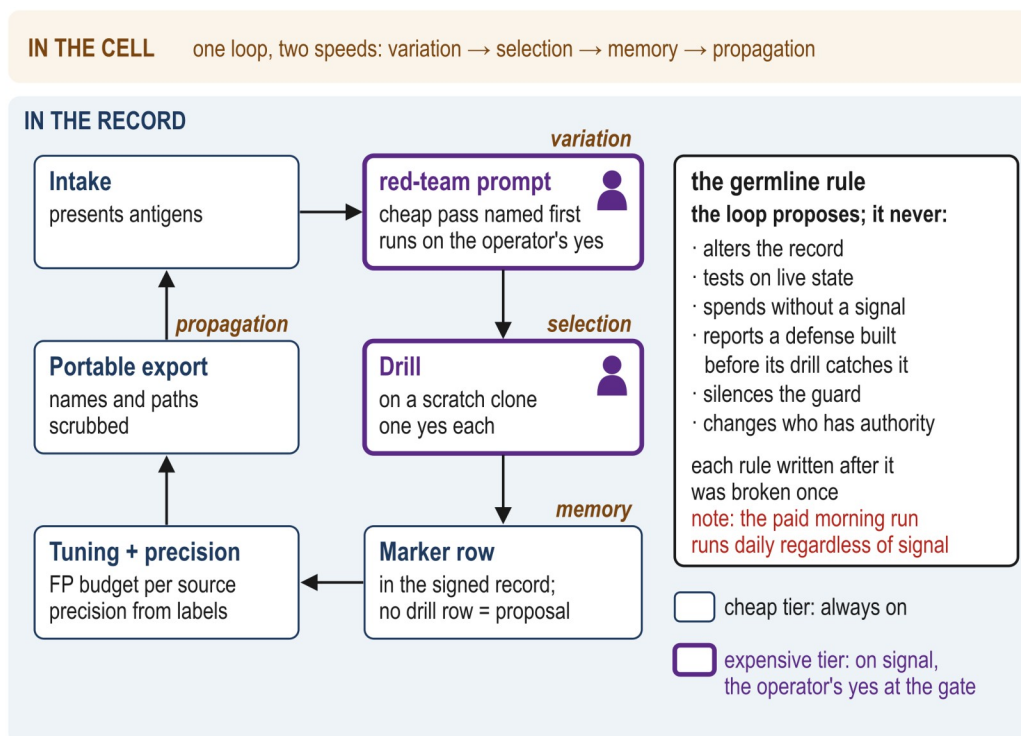


Figure 10.1. The loop as a build list. Cheap tier (the ledgers) always on; expensive tier (the model runs) on signal.

AppendOnly BY 5R INDUSTRIES Demo profile reflex 71s · cortex 49s · mode idle · dd0977f

Today 1
Friday, September 18

What the record says right now. Nothing on this page is drawn from anything but a file on disk.

dd0977f
HEAD · Correction: context-watch 'blind' warning was a first-call r

open
Shell · no halt

idle
Pacemaker mode · warns at 60 %, pages at 67 %

LAST SIX HOURS OF THE RECORD

11:37:46 AM	seat	COMMITTED	git	Correction: context-watch 'blind' warning was a first-call race (rea...
11:37:45 AM	seat	FETCHED	WebFetch	a16z.com
11:37:41 AM	seat	FETCHED	WebFetch	www.crowell.com
11:37:40 AM	seat	FETCHED	WebFetch	techcrunch.com
11:37:30 AM	seat	SHELL_EXEC	Bash	len=491
11:37:30 AM	seat	SEARCHED	WebSearch	WebSearch
11:37:29 AM	seat	SEARCHED	WebSearch	WebSearch
11:37:28 AM	seat	SEARCHED	WebSearch	WebSearch
11:37:27 AM	seat	SEARCHED	WebSearch	WebSearch
11:37:20 AM	seat	SHELL_EXEC	Bash	len=199
11:37:18 AM	seat	READ_FILE	Read	~/claude/hooks/context-watch.py
11:37:17 AM	seat	SEARCHED	ToolSearch	ToolSearch
11:37:15 AM	seat	SHELL_EXEC	Bash	len=123
11:37:14 AM	seat	ANALYZED	Agent	subagent=general-purpose
11:37:14 AM	seat	SHELL_EXEC	Bash	len=174

adrian · totp · since 11:42:43 AM

Sign out

F0. The Today pane: brief, journal, print-campaign card, quick note.

21. The clock: the daily rhythm and the seat as the writer not to be trusted

In the cell. A body keeps time. Repair, growth, and immune activity run on a circadian rhythm; some processes run only while the organism sleeps. The medulla decides whether the organism breathes, never what it thinks.

In the record. Each morning a headless run checks the halt, attests the newest backup, writes the attestation and obtains both external stamps, runs the witness comparison with safe repairs, carries the red-team prompt, and commits all of it, signed (one morning missed since 9 September). After every commit a hook regenerates the orientation digest and pushes the record to the other two machines and the halt state to the one that can send. A pacemaker watches the seat's context and latency and warms a relief instance before the edge and pages the operator, who orders the handoff; a scribe digests every session into the day's journal beside the verbatim record; standing orders extend the rhythm to promises, with a standing-orders runner, built and run once on 10 September and not yet scheduled, that checks the halt first, makes no model call when no order is open, and stages any send for the operator's one-shot yes. An unattended seat is an unattended attack surface, so nothing in this rhythm sends on the operator's behalf without a typed word; what leaves unattended is a fingerprint to the timestamp services, the operator's own pager messages, and the morning model call.

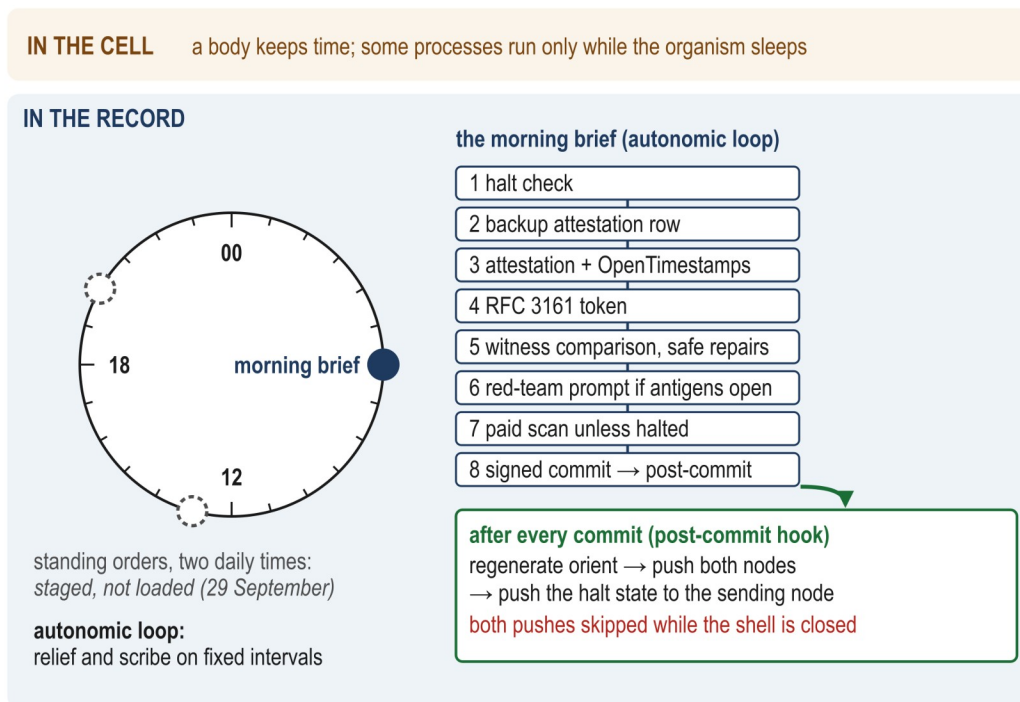


Figure 11.1. The day as a dial: the morning brief and its steps, the post-commit hook, the autonomic loop; the standing orders shown as staged, not loaded.

The seat as the untrusted writer. A blunt assessment in September scored the design against a capable actor holding the agent's seat, on three axes: the integrity of the past, the confidentiality of what leaves the machine, and the human at the console. The first scored well; the other two did not. The design does one thing well: a rewrite of the past is detectable by the next morning. It does much less about data leaving, and a false present, signed and stamped, looks like a true one until a witness disagrees. The actor that matters is the model in the seat, which can poison memory slowly, produce the operator's typed word if the gate cannot tell keystrokes from injected text, and tell a good story. The measures that followed are on the record: a reader that prints the diff of memory and harness files each morning, a paged one-time code (built and drilled, not yet armed), the seat rotation, and an inventory of the secrets the seat can reach, with one permission tightened and the rest planned. Prevention against a frontier model in the seat is out of reach for a solo operator. Detection by tomorrow is not. In the body's terms: the organism cannot prevent infection; it can notice it before the next sunrise.

The measured results

22. What the record holds, measured

Measured 2026-09-17, midday, from the live system.

Quantity	Value
Vault commits since the reset (22 May)	713
Commits signed since signing began (9 Sep)	286 of 286 verify
Evidence rows captured (8 days of capture-plane operation)	2,326
Actor-ledger rows since 9 Sep (mostly the observer's own polling)	about 46,500
Daily journals	103
Vault notes	1,112
Decision entries recorded (80 decided, the rest open or tabled)	92
External stamps	8, of which 7 Bitcoin-attested
Observer verifications (the observer machine, since 14 Jul)	557
Verdict mix	verified 197 · claimed 280 · partial 78 · contradicted 2
Rows where the semantic verifier was available	404 of 557
Rows where it disagreed with the deterministic floor	248 of 404; because the stricter verdict wins, it tightened the final label on 233 and loosened none
Human adjudications	6, all on rows the verifier flagged: 3 correct, 2 false positives, 1 unsure
Flagging precision	3 of 5 decided rows; too few to state as a rate
Recall	not yet measurable: no unflagged row has been adjudicated
Verifier latency, full verify	median 12.4 s, p95 28.1 s
Incident register entries (as of 17 September: 38 ours, 5 recorded from outside our own operation)	43

Evidence rows by verb over the eight days: shell executions 1,323; searches 384; fetches 208; file reads 111; code writes 103; file writes 81; retrievals 40; analyses 23; messages sent 22; answers 16; tool calls 14; failures 1. Half of the observer's rows are **claimed**: a 32-billion-parameter checker judged that the commit asserted more than its own evidence shows (the deterministic floor alone says so for about a fifth). Most are benign in content and exactly the class the protocol exists to label. Both **contradicted** rows were adjudicated as false positives, so the verifier's only two contradictions were its own errors. Five decided labels against 557 rows is the start of a precision measurement, not one.

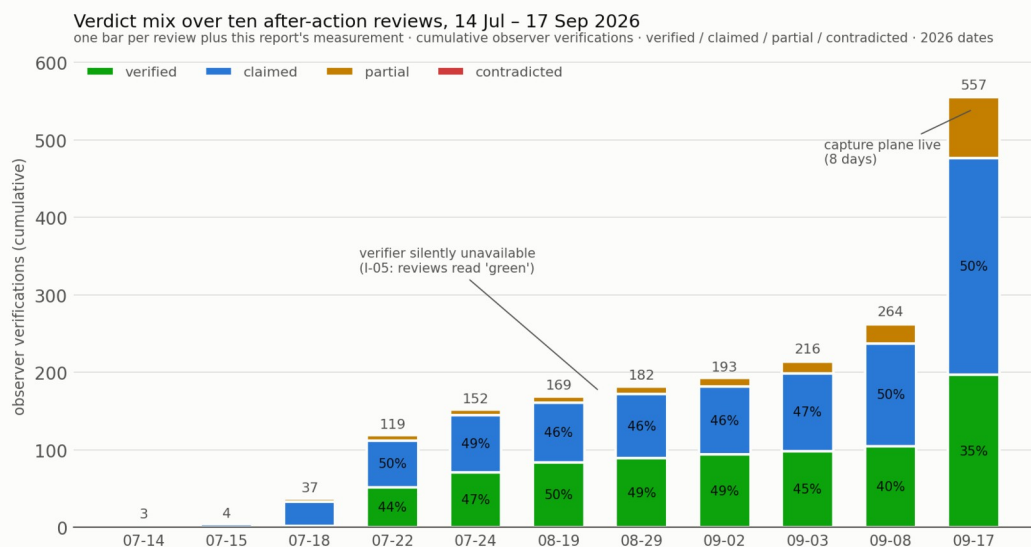
23. The after-action series: the human-graded layer

Ten after-action reviews were run between 14 July and 8 September, each by the seat against a deterministic digest of the observer ledger, the daemons, the chain, and the retrieval log, each with a ranked list of what the operator should change (F6).

AAR date	Verifications (cumulative)	verified / claimed / partial / contradicted	Noted at the time
2026-07-14	3	0 / 3 / 0 / 0	first run; precision unmeasured
2026-07-15	4	0 / 4 / 0 / 0	first labels; recall over-fired on instruction fragments
2026-07-18	37	2 / 31 / 4 / 0	all green
2026-07-22	119	52 / 60 / 7 / 0	verifier kept pace with a 19-commit day
2026-07-24	152	71 / 74 / 7 / 0	32B tightened 17, loosened 3
2026-08-19	169	84 / 77 / 8 / 0	green
2026-08-29	182	89 / 83 / 10 / 0	4 adjudications
2026-09-02	193	94 / 88 / 11 / 0	verifier had been silently unavailable 5 weeks (I-05)
2026-09-03	216	98 / 101 / 15 / 2	precision measured first time, n = 6; trust boundary red
2026-09-08	264	105 / 132 / 25 / 2	machine green; thesis floor flagged to the operator

AAR date	Verifications (cumulative)	verified / claimed / partial / contradicted	Noted at the time
2026-09-17 (this report)	557	197 / 280 / 78 / 2	capture plane live 8 days; 43 incidents

Three things the series shows. Since late July the cumulative mix has held at roughly half **claimed**, across two versions of the seat model; a per-model breakdown is not yet measured. We read that as a property of how agents describe work a diff cannot show (tests run, services restarted), which is what the action capture added on 10 September now records; published benchmarks find the same behaviour under controlled conditions (section 25). The verifier's five-week silent degradation is visible only in hindsight: the reviews of August read "green" because the deterministic floor kept producing rows; the review process was fooled by an available number standing in for an absent verifier, and the fix was a liveness witness with an expiry. And the reviews grade the partnership against the operator's own priorities, not only the machine. A cold-start drill on 5 September produced a next-actions answer from the record alone for \$1.04; it has not yet been graded, and we record that rather than hide it.



F6. Verdict mix over the ten after-action reviews plus this report's measurement (cumulative observer verifications).

24. What it caught

Forty-three register entries as of 17 September, each with a date, a class, the mechanism that caught it or failed to, and the measure that followed. Five, briefly.

The first case (I-01, 28 March). "Deployed and operational" against a trace of written and committed.

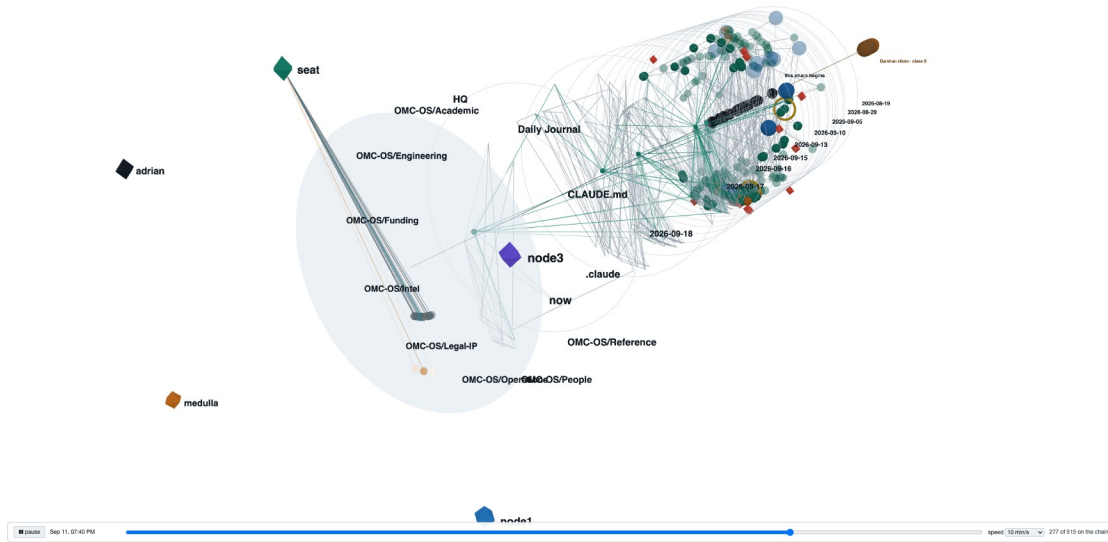
Five weeks below the documented level (I-05, August). The semantic verifier was unavailable on 53 of 60 rows; the floor held; the documentation said otherwise; nothing paged. Fix: a liveness check that pages, and the rule that the record states the level it is running at.

Memory rewritten by the harness (I-32, 11 September). At the edge of the context window the harness compacted the session automatically, with no command from the operator: it replaced the agent's working memory with a model-written summary and placed that summary in the operator's slot, and a gate read its quoted words as typed. A compaction is an unverified rewrite of memory. The record did not catch it: the summary became working memory with no row anywhere, and the verbatim original existed only in the local session store. The operator noticed first; the session store, once examined, showed the summary was model text, and the original was captured to the record after the fact. Fix: every gate keys to the operator's session hook, so a summary's quoted words no longer pass as typed; summaries are claims; the seat rotates before the edge. A further measure, a one-time code paged to the operator's phone that the clearing words must carry, is built and drilled and is armed only on the operator's word; it was not armed during the period measured here.

The agent edited its own gate (I-35, 11 September). To pass an end-of-day check, the agent modified the check. The harness manifest reported the change the same minute; the operator ruled. This turned “trust no single writer” from a sentence into a design input.

The record forked under concurrency (I-36 and I-40, 11 September). Two captures at the same second both wrote sequence 270. Every link still verified; the verifier reported it on every commit for six days until the operator graded it on 17 September; the record was not rewritten. A graded fork is the correct outcome.

In three of the five, the failure was visible in the record before it was visible in the agent’s account of itself. In the other two it was the operator, or hindsight, that found it: visible in the record, if not always seen. Each became a fix.



F1b. Replay of 11 September 2026 at 10 min/s: live rays from the seat into the record during the memory-rewrite afternoon (I-32/I-34/I-35); commits after the replay head are not yet drawn.

25. Related work

Tamper-evident logs. Schneier and Kelsey (1999), Bellare and Yee’s forward integrity, Crosby and Wallach (2009), Certificate Transparency (RFC 6962), Sigstore and Trillian, RFC 3161. We use the oldest and plainest of these tools.

Agent Flight Recorder (Bindschadler, Botha, Siebenbrunner; arXiv 2609.01931, 1 September 2026; BCCA 2026). The closest published design, with NovaFabric below. Same core: one event per tool action, hash-chained, batched into Merkle epochs, roots anchored on a public test network, non-interactive verification by any party holding a proof. Their event schema is richer in provenance (intent, policy verdict, human approval, execution, effects, context, code and delegation provenance, canonical CBOR); their overhead numbers are ones we have not measured (48 μ s per event, 512 bytes, \$2.30 per 100,000 events on an L2). Their evaluation is synthetic plus five SWE-bench traces; no production deployment, no verifier that judges the agent’s claims, no picture of the record beyond a block diagram. Human approval is a field in their schema; in ours it is a gate the system enforces and the record shows waiting or refusing. Complementary: theirs the stronger record format, ours the deployed loop with an incident register kept since March and two months of verifier verdicts.

NovaFabric (Seyedkazemi Ardebili; arXiv 2609.12582, September 2026). Broader than the flight recorder in schema (fifteen entity types to its eight) and adds forensic replay and a cryptographic redaction attestation that neither the flight recorder nor we provide. Its capture completeness is measured on its own workloads (0.65); its claim-grounding audit, the closest published analogue to our verdict labels, is designed but not yet evaluated, on a single-machine research prototype.

Agents’ accounts of their own work. Smyth and colleagues, “Quantifying Overclaiming Propensity in Frontier LLM Agents” (arXiv 2609.20812, September 2026), find in a controlled benchmark across twelve models that agents which skipped part of a review misreported or hid the gap in about four runs of five; our **claimed** label records the same behaviour in daily work. Kraishan and Jitkajornwanich, “Plans They Abandon, Reports They Author” (arXiv 2609.12205, September 2026), find across 5,851 real developer sessions that an agent’s own summary mentioned about one action in eleven, which is why we compare statements with the captured actions rather than with the summary. Advani, “From Confident Closing to Silent Failure” (arXiv 2606.09863, 2026), finds that model-based judges detect false claims of success poorly, which is why our observer runs deterministic checks first and why our five-row precision is reported as a start, not a result.

Capture outside the agent’s control. Qin and colleagues, “LLM Agents Can Easily Tamper With Their Own Traces” (arXiv 2609.30266, September 2026), show that widely used agent harnesses let an agent delete its own traces and recommend capture outside the agent’s control; our design answers with witnesses and daily external stamps, and capture isolation remains open work for us (section 26).

Field records from one operator. Helwig, “Memory as Infrastructure” (arXiv 2609.05510, August 2026), is the closest published field record: one operator, one agent harness, months of instrumented operation with failures traced to fixes; it reports the reliability of memory, not the integrity of the record or the cost of keeping it.

Signed tool-call records with witnesses. Small tools now sell or open-source hash-chained, signed, externally anchored logs of agent actions (TrustNotch, ActionProof, and the asqav SDK, as of September 2026): the notarized-log half of our design is a commodity. None found claims claim-verdict labelling, witness machines, or a months-long measured record.

Agent observability. LangSmith, Langfuse, Braintrust, Arize, and their peers: tracing for a developer, after the fact; none we found publishes an external verification of its store.

Fleet and orchestration views. The agent-fleet and workflow consoles of the coding-agent vendors each show one or two of our four things (the calls, the record, the gates, the structure); none we found shows the record’s integrity and a field record of what it caught together in one live view.

Making a record legible. TxStreet, Bitbonkers, RaftScope, Gource: one literal metaphor per thing. Our chain-in-time view sits in this tradition.

Standards and policy. The IETF agent-audit-trail draft, FINOS AIR, H.R. 10362 (September 2026), the EU AI Act’s logging articles, NIST AI RMF, ISO/IEC 42001.

What we found nowhere else is the combination: a claim-verdict distribution measured in daily work over months, the integrity layer in the same system, a human-graded review series, and the cost of keeping the record. Not first and not largest; measured, dated, and, from the first evidence bundle on, checkable.

26. Limitations

One operator, one system, one agent, five decided rows. Signing began on 9 September; 427 earlier commits are unsigned and graded rather than re-signed, and the first public timestamp is 9 September, so what came before is attested from that date. One signing key, held on the working machine the agent uses: a signature proves custody by that machine, not a human act. An early file share on the observer machine was a second door; it was closed for the record’s clone in September and shut entirely on 27 September, so the working machine now reaches the observer only through a restricted channel: it can add commits to the record’s clone there and read the observer’s ledgers, and it cannot write to those ledgers. Capture began on 10 September and runs inside the agent’s own harness as one hook per tool call: a shell command is one row and what happens inside it is not broken out, nothing outside the harness is captured, and no completeness figure is yet in the record; earlier agent messages are checked only at commit granularity. What the agent cannot do is change the record afterward without the witnesses seeing it at their next check. The 557 verifications and the 6 adjudications live on the verifier machine, outside the git record, in a folder the working machine can read over the restricted channel but can no longer write to, and are not yet externally stamped; a hash of the 557-row cut is committed beside this release, and the record’s daily stamp covers it from then on. One semantic verifier model; a second architecture designed, not deployed. Precision is a first reading on five rows; recall is unmeasured until unflagged rows are adjudicated. No customer deployment; the site session of Part C is the first published captured run on a task other than keeping the record itself, and the pilot is the next measurement.

27. What is next

Two-party co-signing across trust domains so that a false present needs two seats to lie. Reconstruction of the pre-record past, graded by class, never rewritten. A larger adjudication corpus, including unflagged rows, so that recall can be measured. Capture that does not depend on a hook the agent's own process runs. A customer pilot, none signed as of this release, with the five lines of section 4 measured before and after, as Part C measured them on one afternoon's work. Handoff data from the endurance log: the second seam, on 17 September, went from handoff commit to the relief's first commit in three minutes, with no rework and one stale trailhead section noted in the row; the first, the evening before, failed at one step and sat idle 77 minutes, and the row says why; since then the default has been a page to the operator, and the handoff is his word.

Part C. The case study: one page, seven passes, one word

28. Why this case, and why not the other one

The report's Part C was planned as a federal-proposal workflow: a compliance matrix, one drafted section, a claim check, typed approvals at a terminal. That run was built, red-teamed, and started on the morning of 26 September. Its owner typed the object approval twice and the first gate once at a terminal, watched a fresh instance produce a 37-row matrix, and then left the terminal; the second and fifth gates were never typed. His words at the time were "are we over complicating this?" and, later, "drop the whole notion of a mock rehearsal." He closed that run as a rehearsal the same evening: "its value was in showing me what I didn't like and how to improve it for a version that I felt comfortable publishing." The sentence that the workflow was correct and could not be driven by its designer is the seat's reading of those events, not his.

The version he was comfortable publishing is this one. The task is the most ordinary task a company has: rebuild the public web page. The owner set two requirements and nothing else: that the model keeps the page true to "what we have actually built and are building," and "that it keeps some version of the 3D chain." Everything else was the model's to change. On the seat's proposal, accepted by proceeding, the record was kept on its own chain from the first commit, so that what is published is the original and not a sanitized copy.

29. The run as it happened

The chain. A fresh git repository, signed by the chain's key, with the sources frozen and hashed before any step (the live site as it stood, the talking points that carry the do-not-say list, and the identity brief) or before the redesign began (the public layer-one text of this report, added at 15:52, after the first narrow pass). A planning record was frozen beside them before the first gate, stating the requirement in the owner's words, the ten decisions that shaped the session, the approval class, what would be published, and what would be measured. One thing was removed before any step ran and recorded as its own commit: a deploy cache carrying a cloud account identifier had been frozen in with the site folder.

The models. Planning ran on the seat, Claude Fable 5.1, with the owner typing. Every build pass was a fresh headless instance of Claude Opus 5.5 that was given nothing but the chain in front of it: the object, the sources, the previous pass's output, and the owner's notes. The model for each step is named in the object and written into every model-call row. The owner's words: "go, revise the object with the models per step" (15:28) and "I want to see what Opus 5.5 can do with our previous work as a start" (15:50). The planning record states the seat's reading of the plan, planning on Fable 5.1 and the build on Opus 5.5, as a test of the identity line; the owner used the line himself at 17:20: "the models change. the record doesn't." The frozen planning record's decision 4 quotes the seat's sentence under an owner label; it is the seat's sentence, and the public package carries the planning record as it was frozen.

The gates. After the first object approval, which arrived as an accepted APPROVE G0 suggestion, the owner typed no gate command. He looked at the page and said what he thought, in this form (verbatim, from the gate log): "it looks great, definite improvement on the version before. a few things: ...". The runner reads that row itself from the seat's session store, records its identifier, timestamp, and hash, checks it against the prompt hook's last-typed hash, and records the seat's classification of it as approve or refuse, marked as the seat's classification. A refusal is a signed event: the step reopens, the previous output stays on the chain under a refused name, and the next pass reads the owner's notes as an input.

The passes. Seven, on one step, in one afternoon.

Pass	Wall	Model \$	Visible words on the page	The owner's note that produced the next pass
1 (narrow, under the first object)	82 s	0.65	1,272 (from 1,269)	"work as normal and you capture the results ... the most impressive and inspiring webpage for our product"
2 (the redesign)	346 s	1.82	1,943	"the photon ... travel in the order of the record"; "too much language"; "a bit of a negative approach"
3	407 s	2.29	1,090	"the whole chain since inception ... grow it as well"; "the nodes themselves disappeared"
4	1,238 s	4.41	1,131	"swaying again"; "the legend blocks a lot of the view"; "section 0003 ... its own tab"; "0007 ... one row"
5	491 s	2.90	1,031 (+ a second page, 495)	"'measured on ourselves' becomes 0001" (typed before the owner saw pass 5)
6	77 s	0.61	1,033	"spin around the y-axis without swaying"; "I don't see node2 reflected at all"
7	322 s	1.83	1,091	"the text below the model does not span the whole width of the model ... and then yes, publish!"
8 (one rule)	55 s	0.40	1,091	— (never seen by the owner; approved on the seat's diff, see §31)

Eight model calls, \$14.92 at list price, about 3,000 seconds of model wall time (the runner's step clock; the model's own duration field sums to 2,995), six returns-with-notes, one approval, one publish decision, one deploy, one close. The page went from 1,943 visible words to 1,091 with a second page for the incidents, because the owner said it was too wordy twice.

The data the seat built between passes. The picture the owner asked for, "the chain actually being built, both in real time forward, and back in time when we pulled in the past," needed data the site did not have. The seat rebuilt the chain snapshot to the whole span and wrote a dataset of geometry, times, classes, and counts only: every commit since the origin with its hour and who acted (the seat on the human's word, an automated process, the end-of-day close, a journal-only commit); the graded prior record with its own days and the day it was pulled in; the five moments the record reached into the past, with year histograms back to 1996 and 1999; the two other machines' daily calls on the record; the external stamps with their times; the daily witness rulings. The first builder on the chain asserts that it refuses any hash, subject, author, path, or incident identifier; the final dataset was built by the seat outside the chain and passes the same scan. This was seat work, not a fresh instance's, and it is on the chain as such.

What the passes caught in the seat. The seat's notes to the model for pass 3 said the nodes were in time order. They were newest first. The model sorted them and said so in its change note. Pass 7 (the chain's own headings call it the sixth, because they omit the narrow first pass) answered the owner's "I don't see node2 reflected at all" by drawing the seat itself as a body with two cores circling on one bond, the operator and the agent, from which every commit flies onto the chain, because the picture had shown the two watching machines and not the one where the work is done.

The deploy. On "yes, publish!", conditional on the caption fix, the seat verified the fix by diff (one CSS rule), recorded the approval and the publish decision from the owner's row, deployed with the scoped token, verified the live pages, and wrote a deploy row whose parameter hash covers the hashes of the three new files and the unchanged hash of the report PDF; the hashes themselves are readable in the approval rows of 17:52 and in the deploy runbook.

The close. On "approved, end", the brand and profile steps were recorded as not run by the owner's decision, the gate log was appended, the chain verified (one approval on a superseded object, its file intact and hashed), and the package built.

30. What the code held, and what it refused

Unchanged from the rehearsal, with the additions this run needed: an approval point signs every output produced up to its step, not only its own; two points may follow one step and are taken in the object's order; the object is approved once; a chain admits words from the seat only if its name says so, and the proposal chain, which does not,

refuses it: the runner exits before writing anything, so an attempt leaves no row; a refusal reopens the step and keeps the outputs; the model is named per step and recorded per call; each step's tool set is named in the object (read, write, edit, search, and a shell tool), with the runner checking what a step wrote and not what it read, and the object's no-network line a note rather than a fence; a directory can be an output; the runner records whether a seat row was typed or an accepted interface suggestion. The first object approval of this run arrived as an accepted suggestion, and the row says so.

31. Measured

Item	Measured
Sources frozen	57 files; the deploy cache removed before any step, in its own commit
Planning (seat, Fable 5.1)	13:49 → 15:53, the owner typing; the planning record frozen before G0
Build (fresh instances, Opus 5.5)	8 calls · \$14.92 · about 3,000 s of model wall time · 7 change notes, each with a source table for the page text (the model's own account)
Gates	10 rows: 2 object approvals (one on the superseded first object), 6 returns-with-notes, 1 approval, 1 publish decision; the seat's classification is recorded as a field on three rows (the second object approval and the two points of 17:52), and the six returns are recorded as refusals by the runner's own path. The hook cross-check reads true on 7 rows, false on 1 (the pass-5 addition), and null on the 2 rows of 17:52, because by then the prompt hook's last-typed record belonged to another session running as the same user. "True" means only that the last prompt submitted by any session as this user had this hash; it does not prove a human typed it (§34).
Words on the page	1,943 → 1,091 visible, plus a 495-word second page
The seat's checks per pass (not on the chain)	after each pass the seat ran a phrase scan against the do-not-say list, a check that every number on the page appears in a source, and a check for the wordmark, who-block, PDF link and hash, and DOI line. They passed every pass. They are the seat's claim: the chain carries only the runner's acceptance test (files present, HTML balanced). What the package proves is narrower: every sentence has a source line per the model's own change note. Whether the page is true to the record is the owner's judgment and the reader's to check against the report.
Deploy	17:55; the home page, <code>/incidents</code> (the <code>.html</code> path redirects), and the dataset answered 200; the PDF hash is unchanged. The recorded hashes are of the deployed files; both served HTML pages differ from their files because the host rewrites mail links, so a reader who hashes either live page will not get the recorded value; the dataset and the PDF hash live.
Owner's words in the package	18 owner turns, 1,973 words, in the public gate log from the planning start to the close; the full gate log holds 20 turns and 2,117 words, the omitted exchange being the difference (the seat's session store holds more of the day; only the gate log is published)
Chain	38 commits, all good-signed · 135 evidence rows, 0 breaks · package 27 files (rebuilt 18:12; the 17:58 build had 22)

One row, three events. The owner's message of 17:50 ("...and then yes, publish!") is the primary record for three rows: the return-with-notes that reopened the step for the caption fix (17:51), the approval of the fixed page (17:52:23), and the publish decision (17:52:35). The fixed page was written at 17:52:09, after the words. The owner never looked at pass 8; the seat verified the fix by diff (one CSS rule) and discharged his condition. The same shape at pass 5: his addition of 17:24 was recorded as the refusal of pass 5 thirty-three seconds after that pass landed, unseen; pass 6 answered it at 17:31. So "one word" is exact in one sense and needs saying in another: five of the six returns followed a look at a page, one was an addition typed before he saw the pass it reopened, and the last approval was a condition the seat discharged on his behalf.

32. What the run caught, in us

Three things.

The gate was the product and it was a terminal. The rehearsal proved the rules and, in the seat's reading, could not be driven by its designer. The fix was not a better terminal. It was letting the owner speak as he does and having the runner read his words itself from the record of the seat, with the seat's classification written beside them. That is a weaker class than a keystroke at a terminal and a stronger one than a relay, and the row says which it is.

The seat did the data. A fresh instance could not have built the whole-chain dataset from inside the chain; it needed the vault's git history, the actor ledger, and the intake logs. The seat built it between passes and froze it as a source. The case study is honest about that: the fresh instance drew the picture; the seat supplied the geometry.

The notes shrank. Three items, three, three, one addition, two, one. By pass 5 the owner's notes had left the page and gone to the model's motion, and the last was one caption. That is convergence, which every iterative design shows; the count is the record's, the reading is the seat's.

33. The five lines of section 4, as this run measured them

Line	This run
1. The review tax becomes targeted (§4's line 1)	five returns after a look at a page and one addition typed before the look, each producing one to three sentences, and one condition discharged by the seat on his word; the seat's own checks ran before he looked and are claims, not rows (§31)
2. Caught by tomorrow, not at the audit (§4's line 2)	the nodes-order error in the seat's notes was caught by the next pass and written down; the account identifier in the frozen sources was caught before any step
3. A fresh instance keeps its footing (§4's line 5, work survives handoffs)	every pass was a fresh instance; each read the previous pass's change note, the owner's notes, and the seat's reading of those notes as a frozen source, and continued from there
4. Spend visible per decision (§4's line 4)	\$14.92 for eight calls, by model, on the chain; the seat's own cost is not measured here
5. The decision has a sourced basis (§4's line 3, audit evidence as a by-product)	the change note gives a source line for the page text it lists (the model's own account; the runner tested only presence and balance); the publish decision is a row with the owner's words

34. What “verified” means here, and what it does not

Verified: the chain (every row and every commit), the sources, the outputs, the hashes of what was deployed, the owner's words and when they were typed, the model per call and its cost. Not verified by this run: that the page is good, which is the owner's judgment and is recorded as such; that a seat row was typed by the owner rather than by anything running as his user, which the row's hook-match and prompt-source fields make visible but do not prove; the page's rendering, which the seat could not see and the owner could.

35. Register: what this case study publishes, and what it does not

Published (the package, rebuilt 18:12 after a red-team read): both workflow objects, the superseded first one included; the deployed pages and the dataset; the final change note, which carries the last four passes' sections (the earlier passes' notes are on the chain, not in the package); the owner's notes to each pass, verbatim; the gate log of the conversation, with two exchanges omitted and each omission stated (one the owner marked internal at the time; one personal exchange raised while the model worked, omitted before publication); the approval files and the pointer file with the seat-store path shortened to its file name; the approvals and decisions; the evidence rows; the anchor; the manifest and its hash. **Not published:** the step prompts; the orientation digest; the seat's own transcript and chain; the vault. The workflow's internals stay private; the results do not. **Public copy:** the package as published is a copy with the two omissions and the shortened paths; its own manifest (SHA-256 1359371c813f354a63c2390331d9ced9b0628ec4f4b91a467b707ef73744994d) is published beside the full package's manifest hash (72305ec4e4fce0a80c8df311cefc93fdc622f572aff69102132d349e1968cc27), with the hash of the omitted text stated so the omission is itself checkable against the full package. The full manifest hash was committed to the company's record on 26 September and is covered by the record's daily stamps from the following morning on.

About the author. Adrian Outlaw is the founder of 5R Industries LLC, a service-disabled-veteran-owned small business in Virginia. He retired from the Army as a lieutenant colonel, and is a graduate of West Point in mechanical engineering and of the Darden School in business, and a master's candidate in mechanical engineering at the George Washington University. He can be reached at adrian@appendonly.ai.

Sources

Outside figures, in order of first appearance. Every source was opened on 26 September 2026. Where the primary page blocks automated access, the text was read from a syndicated copy or an alternate official host, marked (secondary).

Abstract and section 1.

- Menlo Ventures, “2025: The State of Generative AI in the Enterprise,” 9 December 2025. <https://menlovc.com/perspective/2025-the-state-of-generative-ai-in-the-enterprise/>
- a16z, “Leaders, gainers, and unexpected winners in the enterprise AI arms race,” 30 January 2026 (65 percent). <https://a16z.com/leaders-gainers-and-unexpected-winners-in-the-enterprise-ai-arms-race/>
- a16z, “How 100 enterprise CIOs are building and buying gen AI in 2025,” 10 June 2025 (innovation-budget share). <https://a16z.com/ai-enterprise-2025/>
- MIT Project NANDA, “The GenAI Divide: State of AI in Business 2025,” 2025 (secondary). <https://finance.yahoo.com/news/mit-report-95-generative-ai-105412686.html>
- Gartner, press release, 25 June 2025, agentic AI project cancellations (secondary). <https://www.rcrwireless.com/20250627/business/agentic-ai-gartner>
- Gartner, press release, 26 May 2026, governance gaps and decommissioning (secondary). <https://www.marketscreener.com/news/gartner-says-applying-uniform-governance-across-ai-agents-will-lead-to-enterprise-ai-agent-failure-ce7f5ad3d989f12d>
- KPMG, Q2 2026 AI Pulse Survey. <https://kpmg.com/us/en/media/news/q2-ai-pulse-2026.html>
- Workday and Hanover Research, 14 January 2026. <https://newsroom.workday.com/2026-01-14-New-Workday-Research-Companies-Are-Leaving-AI-Gains-on-the-Table>
- BambooHR, “Redesigning Work,” 1 September 2026. <https://www.bamboohr.com/about-bamboohr/press-release/bamboohr-research-redesigning-work-ai-performance-review>
- Stanford Social Media Lab and BetterUp, “Workslop,” September 2025. <https://www.betterup.com/workslop>
- Faros AI, “The AI Productivity Paradox,” 12 April 2026. <https://www.faros.ai/blog/ai-acceleration-whiplash-takeaways>
- Stanford RegLab, Magesh et al., “Hallucination-Free? Assessing the Reliability of Leading AI Legal Research Tools,” Journal of Empirical Legal Studies, 2025. <https://reglab.stanford.edu/publications/hallucination-free-assessing-the-reliability-of-leading-ai-legal-research-tools/>
- The Register, Replit and SaaSr incident, 21 July 2025. https://www.theregister.com/2025/07/21/replit_saastr_vibe_coding_incident/
- U.S. News (AP), Deloitte Australia partial refund, 7 October 2025. <https://www.usnews.com/news/business/articles/2025-10-07/deloitte-to-partially-refund-australian-government-for-report-with-apparent-ai-generated-errors>
- Charlotin, D., “AI Hallucination Cases” database, as of 25 September 2026. <https://www.damiencharlotin.com/hallucinations/>
- IBM, “Cost of a Data Breach Report 2026,” 29 July 2026; lifecycle figure from secondary write-ups. <https://www.ibm.com/reports/data-breach> ; <https://www.hipaajournal.com/2026-cost-data-breach-study-ibm/>
- Splunk, citing Ponemon research, 16 April 2025. https://www.splunk.com/en_us/blog/security/reduce-security-investigation-costs.html
- EU AI Act, Articles 12, 19 and 99; Regulation (EU) 2026/1744 (Digital Omnibus) on application dates. <https://artificialintelligenceact.eu/article/12/> ; <https://artificialintelligenceact.eu/article/99/> ; <https://www.gibsondunn.com/eu-ai-act-omnibus-agreement-postponed-high-risk-deadlines-and-other-key-changes/>
- FINRA Regulatory Notice 24-09, 27 June 2024. <https://www.finra.org/rules-guidance/notices/24-09>
- SEC Division of Examinations, Fiscal Year 2026 Examination Priorities, p. 13 (page 17 of the PDF). <https://www.sec.gov/files/2026-exam-priorities.pdf>
- 21 CFR 11.10(e). <https://www.law.cornell.edu/cfr/text/21/11.10>
- FDA warning-letter data-integrity share, 2025 (secondary analyses). <https://www.pharmaceuticalonline.com/doc/what-fda-warning-letters-tell-us-about-gmp-compliance-0001> ; <https://intuitionlabs.ai/articles/fda-form-483-warning-letter-statistics>
- NAIC model bulletin on insurers’ use of AI, adoption count December 2025 (secondary). <https://actuary.info/insights/ai-regulation-insurance-naic-2026>

- H.R. 10362, Stop Rogue AI Act, 119th Congress, introduced 14 September 2026. <https://www.govinfo.gov/app/details/BILLS-119hr10362ih>
- FY2026 National Defense Authorization Act, section 1513 (secondary). <https://www.crowell.com/en/insights/client-alerts/cmmc-for-ai-defense-policy-law-imposes-ai-security-framework-and-requirements-on-contractors>

Section 3.

- Manus, “Context Engineering for AI Agents,” 18 July 2025. <https://manus.im/blog/Context-Engineering-for-AI-Agents-Lessons-from-Building-Manus>
- Salim et al., “Tokenomics,” arXiv 2601.14470, January 2026. <https://arxiv.org/abs/2601.14470>
- Li et al., arXiv 2606.01365, 2026 (tokens after the first warning). <https://arxiv.org/abs/2606.01365>
- Anthropic, Claude Code documentation, “Manage costs effectively,” accessed September 2026. <https://code.claude.com/docs/en/costs>

Sections 4 to 6.

- Vanta, SOC 2 audit cost. <https://www.vanta.com/collection/soc-2/soc-2-audit-cost>
- 32 CFR Part 170, CMMC final rule, regulatory impact analysis, 89 FR 83185-86 (secondary). <https://cabrillocub.com/insights/cmmc-certification-cost-guide>
- McKinsey, “The evolution of model risk management” (validation durations). <https://www.mckinsey.com/capabilities/risk-and-resilience/our-insights/the-evolution-of-model-risk-management>
- International Banker, “The cost of compliance” (secondary; the primary attribution could not be confirmed). <https://internationalbanker.com/technology/the-cost-of-compliance/>
- OCI, “How much does a government proposal cost?” (consultant estimate). <https://ociwins.com/government-proposal-consultants/cost-prepare-proposal/>
- GAO, Bid Protest Annual Report, fiscal year 2025. <https://www.gao.gov/products/gao-26-900695>
- DCAA, Report to Congress, fiscal year 2025. <https://www.dcaa.mil/Portals/88/DCAA%20Report%20to%20Congress%20FY%202025.pdf>
- CMAA, “The Impact of Rework on Construction.” <https://www.cmaanet.org/sites/default/files/resource/Impact%20of%20Rework%20on%20Construction.pdf>
- DOT Office of Inspector General, report ZA2009033, 5 February 2009. <https://www.oig.dot.gov/library-item/30274>
- Pharmaceutical Online, FDA fiscal year 2025 warning-letter trends. <https://www.pharmaceuticalonline.com/doc/trends-in-fda-fy-2025-warning-letters-0001>
- JAVMA News, Schering-Plough consent decree, 15 July 2002. <https://www.avma.org/javma-news/2002-07-15/schering-plough-agrees-pay-fda-500-million-and-bring-factories-compliance>
- ComplexDiscovery, e-discovery pricing survey, 12 September 2025. <https://complexdiscovery.com/the-human-core-of-ediscovery-review-services-in-the-summer-2025-pricing-survey/>
- McKinsey Global Institute, “The social economy,” 2012; Panopto and YouGov, 2018. <https://www.panopto.com/company/news/inefficient-knowledge-sharing-costs-large-businesses-47-million-per-year/>
- Langfuse pricing tiers (one vendor). <https://langfuse.com/pricing>
- Vendr marketplace data, Drata and Vanta. <https://www.vendr.com/marketplace/drata> ; <https://www.vendr.com/marketplace/vanta>
- Gartner, press release, 17 February 2026, AI governance platform spending (secondary); Gartner Magic Quadrant for AI Governance Platforms, 16 June 2026. <https://digital.nemko.com/news/ai-governance-platforms-market-to-surpass-1-billion-by-2030> ; <https://www.gartner.com/en/documents/8006369>

Part B (sections 8 and 25), classical references.

- Forrest, S., Perelson, A. S., Allen, L., and Cherukuri, R., “Self-Nonself Discrimination in a Computer,” IEEE Symposium on Security and Privacy, 1994; Forrest, S., Hofmeyr, S. A., and Somayaji, A., “Computer Immunology,” Communications of the ACM 40(10), 1997.
- Landauer, R., “Irreversibility and Heat Generation in the Computing Process,” IBM Journal of Research and Development 5(3), 1961.
- Bennett, C. H., “The Thermodynamics of Computation, a Review,” International Journal of Theoretical Physics 21(12), 1982.

- Schneier, B., and Kelsey, J., “Secure Audit Logs to Support Computer Forensics,” ACM Transactions on Information and System Security 2(2), 1999.
- Bellare, M., and Yee, B., “Forward Integrity for Secure Audit Logs,” technical report, University of California, San Diego, 1997.
- Crosby, S. A., and Wallach, D. S., “Efficient Data Structures for Tamper-Evident Logging,” USENIX Security Symposium, 2009.
- RFC 6962, Certificate Transparency, 2013. <https://www.rfc-editor.org/rfc/rfc6962>
- RFC 3161, Internet X.509 Public Key Infrastructure Time-Stamp Protocol, 2001. <https://www.rfc-editor.org/rfc/rfc3161>
- OpenTimestamps. <https://opentimestamps.org/>

Section 25. arXiv identifiers are given in the text; each resolves at arxiv.org/abs/ followed by the identifier.