How to build it

Aotearoa New Zealand · © John Stroh

Version 0.3 · revised 13 September 2026 · this version as a PDF

If you run agents and want to be able to show what they did, this is the order to build it in, what each part demands, and the two steps that take the most work. It is written from one implementation, and no build time is claimed, because none has been measured.

Implementation guide · Version 0.3 · September 2026

The shape of it

Six things have to exist, and the rest of this guide is about building them in the right order.

1. **Events.** Every action an agent takes, recorded as it happens: who acted, what they did, to what, and where it sits in the sequence.
2. **A seal chain.** Each event hashed and linked to the one before, so a change anywhere breaks the chain.
3. **Attestation.** Batches of sealed events sent to an outside authority that says: this existed by this time.
4. **A mandate.** A written statement of what each agent is permitted to do, recorded before it acts.
5. **An inventory.** The list of agents, so you can tell which ones you meant to have.
6. **A seed.** A number nobody can predict, fixed before a period and revealed after, that decides what gets examined.

The first three give you a record somebody else can rely on. The second three give you something to compare conduct against and a way to look that cannot be anticipated. The first three are worth having alone — many organisations will stop there. What they will not do

alone is catch an agent exceeding its authority: that needs the second three, and the second three need the first.

Start with events, not with cryptography

Begin with the events rather than with hashes and timestamps. The cryptography is the part with published answers; the events are the part that has to be decided.

A system running agents usually logs something already, and that log is usually not usable here, for one reason: it was written for debugging, so it records what the code did rather than what the agent decided. You want the tool call, its arguments, the object it touched and the identity it acted under — not a formatted message about it.

Get this wrong and you will seal a perfect, tamper-evident record of the wrong thing. The cryptographic steps that follow have published answers. The mandate does not: it is the one part of this that cannot be generated.

Two practical points. Record reads, not only writes: an agent that reads a file it had no business reading has done something, and a record of writes alone cannot show it. And hash the **content**, do not store it — the hash is what makes the record checkable, and keeping the content turns your audit log into a second copy of everything you hold, with all the obligations that carries.

The sealer must be unreachable by what it records

The sealer takes events, chains them, and sends batches out for attestation.

The requirement is not that it is fast or clever. It is that **nothing it records can write to it**. An agent that can reach the log, or the signing key, can cover its tracks, and then the record proves nothing about the only party you needed it to prove something about.

“Outside the trust boundary” is the requirement. A separate host is one way to meet it. So is a separate process under a different identity with append-only permissions, and that is the cheaper place to start — a configuration change rather than a procurement, though no time for it has been measured.

This is the one part of the design that prevents rather than records, and what it prevents is narrow. An isolated sealer stops the cover-up rather than documenting it. It does not stop the act: the harm still happens, and what stops being available is denial by rewriting. An act the record never saw stays deniable. Everything else in this guide is evidential.

Attestation: one token covers a whole batch

You need an outside party to attest that a batch existed by a given time. The standard is RFC 3161, it has been in use since 2001, and a token covers an entire batch rather than an event — so the cost does not scale with how busy your agents are.

Two decisions.

Who attests. A provider independent of you. There is a real gap here for New Zealand organisations: the implementation behind this series uses a European provider, free and not qualified, which supplies the evidence rather than the statutory presumption. A qualified service operating under New Zealand law is a business somebody should be running. The offer to help write its specification stands.

How often. The batch interval is not a performance setting. Two events inside the same batch have no independently attested order; their sequence rests on your word. So the interval is the resolution of your evidence. If you batch daily, you can prove things about days and you are the only witness to anything shorter. Choose it against the timescale of the conduct you would need to establish, then pay for that. Choosing it to save tokens fixes the resolution of your evidence to a figure chosen for an unrelated reason — which is what the implementation behind this guide did, and it does not meet its own standard on that parameter.

Make the boundaries themselves unpredictable, drawn from the seed. Otherwise an agent that knows when the batch closes can place its actions to keep their ordering inside your word rather than the authority's.

The mandate is the part that cannot be generated

For each agent: what it may do, to what, under whose authority, and what it may never do. Sealed before it acts.

This is not a technical document and it cannot be generated. It is a statement about how your organisation delegates, and writing it down is what forces the delegation to be decided. Writing it down is what surfaces an agent still running on permissions somebody granted for a demonstration and never revisited.

Write rules over **sets** of agents as well as over each one. The failure that per-agent rules cannot see is the one where every agent stays inside its own authority and the combination is the breach — which is the shape of what happened in July, when around twelve hundred agents found a shared channel and seven hundred acted on it. Whether each act was individually within authority is not established — some of it plainly was not — but the convergence is a breach no per-agent rule can express.

The seed, and why “we publish a hash first” is not enough

You need an examination the agent cannot anticipate. That means the choice of what to look at must be unpredictable to everybody, including you.

The common design is to pick a number, publish its hash, then reveal the number later. That proves you did not change your mind. It does not make the number unpredictable: the party who chose it knew it from the start, and that party is one an outsider may need to check.

Two constructions work. Take the number from an external randomness source whose value was undetermined when you committed. Or split it among several holders under a threshold scheme, with the threshold set above the number of holders you control or could lean on. A threshold scheme whose shares all sit inside one organisation leaves the number in that organisation’s gift, which is the condition it was meant to remove.

Derive a separate stream from the seed for each thing that needs to be unpredictable — which records get examined, when the examination runs, when batches close, when findings are released. Knowing one then tells an observer nothing about the others, and at reveal the whole schedule can be recomputed and checked.

The examination, and why an AI assistant makes it worse

Sampling decides which records get looked at. Recognition decides whether looking achieves anything — the chance that a reviewer handed a record containing an improper act actually says so. They multiply. With sampled share s and n improper acts, the chance of missing all of them is (1 − s·q)^n, so a perfect sampling regime staffed by reviewers who do not recognise what they are seeing catches almost nothing.

We measured q once, using machine reviewers: 100 per cent on blatant breaches, 92 on moderate, 58 on well-disguised ones, and no false alarms in thirty-six judgements on clean records. The records, the answer key and the scoring code are published as a bundle so the measurement can be repeated.

The obvious answer is to give a human reviewer an AI assistant. Recent experimental work suggests that makes things worse in a specific way. Shaw and Nave (Wharton, 2026, pre-print) ran three preregistered experiments — 1,372 participants, 9,593 trials — with the assistant’s accuracy randomised. When the assistant was right, accuracy went up. When it was wrong, participants followed it on about four out of five trials and finished **below** where unaided participants landed. Confidence rose either way, and in the study that measured confidence item by item it did not differ between the trials where the assistant was right and the trials where it was wrong.

That is a substitution rather than a second opinion: the reviewer's own recognition does not add to the machine's, it largely replaces it. Their task was a reasoning puzzle rather than an audit record and their participants were not accountable for the outcome, so do not take a number from it. Take the mechanism, and three design consequences:

- **Do not count a human-with-assistant as an independent check.** If you want two independent detectors, the second one must not see the first one's answer.
- **Do not use reviewer confidence as the trigger for a second look.** It did not track correctness.
- **Pay per correct finding and give reviewers feedback on whether they were right.** In the study, incentives plus item-level feedback roughly doubled the rate at which people overrode a wrong assistant — from about one in five to about two in five. It did not eliminate the effect, and in an audit you often cannot supply ground truth. It is still the strongest lever available.

What it costs

Attestation is the cheap part — one token per batch, regardless of volume. Storage is the part that grows, and it grows with reads as well as writes.

The expensive parts are not technical. Writing the mandate takes real time from people who are not engineers. The examination costs whatever a competent reviewer costs, multiplied by the sampled share, and the previous section is the argument for spending that properly rather than cheaply.

No total is offered here because none has been measured, and a placeholder gets quoted back as an estimate.

The order to do it in

1. Record events properly, including reads. Nothing else works until this does.
2. Chain them, and move the sealer outside the reach of what it records.
3. Batch and attest, with an interval you have chosen against the conduct you would need to establish.
4. Write the mandate and the inventory. Expect this to take longer than everything above it.
5. Add split-custody randomness and let it drive the examination schedule and the batch boundaries.
6. Stand up the examination, and design the reviewer's situation as carefully as the cryptography.

Steps 1–3 give you a record that will hold up. Steps 4–6 turn it into something that catches an agent exceeding its authority. Many organisations will find 1–3 worth doing on their own.

The two decisions that cost us most

Both from the implementation behind this work, offered as what went wrong once rather than as a pattern.

The batch interval, set for cost. The consequence does not appear until somebody disputes the ordering of two events, which is long after the decision looks settled.

Key custody, decided late. Who holds the seed, the sealer key and the inventory determines what the whole arrangement actually proves, and it is tempting to leave it until the technical parts work. Decide it first — it changes what you build.

Reading further

The formal statements are in Addendum M. The conformance requirements, including the six-item interface an implementation must expose, are in MIO-STD-01. The argument for why any of this is worth doing is in Cheaper not to look.

Drafted with AI assistance, checked and revised by the author.

▸DISCLAIMER

> **How this was made.** The version number counts drafts of the text. It does not measure the inquiry behind it, which has run over days and across several AI systems, with argument between those systems and within them, directed, refused and repeatedly redirected by the author. The source material was AI-generated, and then adversarially and iteratively refined across a range of tools — systems built by different companies in different jurisdictions, set against each other and against the author. No one of them produced this text, and no one of them reviewed it alone. **The plurality is deliberate rather than incidental.** A single model carries a single set of priors about which sources are authoritative, and this series argues that an evidence base narrowed in exactly that way is how a contested question comes to look settled. Using one model to investigate that claim would have been the claim refuting itself. To name a single model on it would credit that model with work that was neither its own nor done in a single pass. **The plurality was also necessary, and the record should say why.** In drafting, the assisting model repeatedly led with United States institutional sources — a national laboratory, an industry association, a market study nineteen years old — and presented conclusions drawn from them as the state of knowledge. On one occasion European measured data contradicting those conclusions was present in the same research return and was placed below them. Framings were proposed that would have argued against this series' own position using that evidence base, and offered as rigour. Each was refused by the author and the material rebuilt. That is the mechanism these documents describe, occurring in their own making, and it is recorded because a series arguing that evidence bases narrow without anyone deciding to narrow them cannot credibly claim its own production was exempt. The framing, the corrections and the judgements are the author's, and so are the errors. How this site is written sets out what is declared on every piece, who checks it, and where the per-piece record lives.

Cite this version

```
How to build it, version 0.3 (September 2026).
agenticgovernance.digital/downloads/how-to-build-it-v0-3.pdf
```


PREVIOUS
MIO-STD-01 — Attestable Audit of Autonomous Agents

Series contents

NEXT
What we claim, and how sure we are