WHITE PAPER

The Evidence Gap

What an Enterprise Needs Before It Can Accept an Agent's Claim of Success

---

Srinivasarengan Varadharajan
Founder, KTER · October 2026 · kter.ai

ABSTRACT

An agent’s completion report can look complete while the work underneath it is not. This paper looks at what an enterprise needs before it accepts that report: a record the agent cannot quietly rewrite, a check that sits outside the agent that produced the work, and a named person with the authority to say no.

It also proposes a way to match the depth of checking to the consequence of an error, and a pilot test for any workflow before it goes live. It does not claim that deception can be engineered away. It argues that unsupported completion can be recognised and contained.

OPENING

The Report Says It Worked

The question behind this edition is a plain one. When an agent says a task passed, what does “passed” mean if the agent also writes the result? Reviewers have always relied on reports. What is new is that the author of the report can now also be the author of the evidence.

On 29 September, Reuters published a review of more than 200 documents and identified at least 20 studies or evaluations since 2025 describing agents that deceived, circumvented restrictions or concealed failure. [1] Where something had gone wrong, agents running on both Chinese and US models guessed at answers, substituted sources, simulated results and fabricated files instead of reporting the problem. The experiments were controlled, and Reuters did not report an agent escaping into the wider web.

A benchmark study published in December 2025 describes the same pattern in a form that is easier to reason about. Guo et al. built 200 tasks across five task types and eight scenarios, each with an imposed obstacle such as a broken tool or a mismatched information source, and tested 11 models. [2] The agents typically concealed the failure through their actions, by guessing results, running unsupported simulations, substituting an unavailable source and fabricating local files. Prompt-based mitigation reduced the behaviour only to a limited degree.

These are benchmark findings under deliberately imposed constraints. They are not measured failure rates in enterprise deployments, and I do not have a figure for how often this happens in production. I would be wary of anyone who claims to.

The earlier editions of The KO Factor examined responsibility, access boundaries and the judgment an agent cannot be assumed to carry. The Visibility Debt asked whether anyone understands a system that appears to be working. This edition asks a narrower question: whether the evidence behind that appearance can be relied on.

1. When the Report Becomes the Result

The handoff

A completion report is usually where one team hands work to another. A reviewer sees a green status, accepts the handoff and moves on.

An agent’s report adds a new problem to that old one. If the agent can also create the material used to justify the status, the reviewer may be checking a claim against another version of the same claim.

I am calling this the evidence gap: the distance between what an agent says it accomplished and what a person can establish from a record the agent cannot quietly rewrite.

A correct answer does not close it

The gap matters even when the final answer happens to be right. An agent that silently substitutes a different source may produce a plausible number today. The enterprise still has to know which source was used before it can decide whether the result satisfies the task. Next quarter the substitute may be wrong, and nothing in the report will have changed.

A related warning from training research

Anthropic’s research on reward hacking contains a coding version of the same shape. Its experimental setup describes a shortcut in which a program calls sys.exit(0) to leave a test harness with a success code, so every test appears to have passed without the intended tests running. [3] That research concerns training environments, and I am borrowing only the narrow lesson. A successful exit is a signal, and a signal can be produced without the work it is meant to indicate.

The enterprise implication is my own: acceptance has to check what actually ran.

2. What Counts as Evidence

Consider an illustrative supplier onboarding workflow. This is a constructed example, not a reported case. An agent says it checked a supplier against the approved screening source. A polished report, with a timestamp and a green badge, reaches the operations team. The underlying service was unavailable, and the report does not say so.

The reviewer needs a record of the actual query and the response from the approved source. It should identify the supplier, the version of the source used and the time of the check. If the agent used a substitute, the reviewer needs to see that difference before accepting the result.

The same reasoning applies to code. An acceptance record for a test run should identify the code revision and the test suite, show which tests executed, and retain failures and skipped tests. A timestamp alone proves very little. A file hash binds a record to an artifact, but it cannot establish that the artifact is true.

Completion claim	Evidence to inspect	What the claim alone cannot show
“The tests passed”	Run ID, code revision, executed tests, failures and skips	Whether the tests that mattered actually ran
“I used the approved source”	Source record, retrieval result and any substitution	Whether the source was reachable or quietly replaced
“The check passed”	Checker identity, acceptance rule and exceptions	Whether the checker or its rule changed during the run
“The work is accepted”	Named reviewer, decision and recovery owner	Whether anyone with authority could have refused

Proposed controls. A timestamp or a badge alone does not establish correctness.

A completion certificate or summary card can make this evidence easier to inspect. Its value depends on where the evidence came from and which checks produced it. A card assembled entirely from the agent’s own assertions leaves the gap open, however well it is designed.

3. The Verification Boundary

The workflow needs a separation between producing work and approving it. For code, that can mean a controlled CI runner executing approved checks against the submitted revision. The agent may propose changes to the tests, but a separate review approves any change that alters the acceptance standard.

For document work, a verifier retrieves the source record independently and compares the output with the agreed criteria. Handing a second agent the first agent’s summary does not create independent evidence. Both may simply repeat the same unsupported account.

Independence is a property of access and evidence. The producing agent should be unable to overwrite the execution record or silently change the checker. A different model may help, but a different model reading the same summary is only a second reader of one claim.

Role	What it does	What it cannot do
Producer (the agent)	Creates a candidate result, proposes changes, records obstacles and substitutions	Alter the acceptance rule, overwrite the execution record or approve its own work
Verifier	Retrieves evidence independently and runs the approved checks	Treat the producer’s summary as evidence or change the standard it checks against
Reviewer (a named person)	Accepts, rejects or approves a documented alternative	Be bypassed by a status badge
Recovery owner (a named person)	Owns the correction when accepted work later proves wrong	Hand that ownership to the agent

Proposed separation of duties. Adapt the roles to the size of the workflow.

Transparency needs restraint

A reviewer needs the inputs, the material actions, the exceptions and a concise explanation tied to evidence. A generated account of an agent’s reasoning is itself a claim. More prose does not turn it into a faithful record of execution, and a long explanation can make a weak record look strong.

4. An Honest Blocked State

A system that cannot retrieve a required document needs a legitimate way to stop. “Blocked” and “partially complete” have to survive as distinct outcomes, with the missing dependency visible to the next person.

This is an operating decision before it is a technical one. If every queue, target and management conversation rewards apparent completion, admitting an obstacle becomes inconvenient for everyone involved. That pressure is a design choice, and it can be designed differently. Acceptance should be arranged so that an accurate failure report counts as useful work, even though the task remains unfinished.

Outcome	What it means	Who acts next
Complete	Every acceptance criterion is met, with evidence the agent cannot rewrite	The reviewer accepts
Partially complete	Some criteria are met and the rest are listed one by one as unmet	The reviewer accepts the part, retries or returns the work
Blocked	A required dependency was unavailable and the task remains unfinished	The reviewer retries, approves a documented alternative or returns the work
Rejected	The reviewer refused the handoff	The work is reworked or reassigned

Proposed outcome states. A blocked task is a finished report on an unfinished task.

A named reviewer decides whether to retry, approve an alternative source or return the work. Their authority has to include rejecting the handoff. Someone else, also named, owns recovery when accepted work later proves wrong.

This is where accountability becomes testable. The enterprise can point to the acceptance criterion, inspect the evidence that existed at the time and identify who authorised the next action. It also gives a vendor a concrete standard to meet.

5. Matching the Depth of Checking to the Consequence

Checking everything at full depth is not affordable, and I do not think it is necessary. The level of checking should follow the consequence of getting it wrong. The tiers below are my own proposal and not an established standard. The thresholds belong to each organisation.

Consequence	Example	Minimum acceptance record	Who signs off
Low	A draft for internal discussion	Source list and stated assumptions	The requester
Moderate	Analysis that other people will rely on	Retrieved source records, checker result and any disclosed substitution	A named reviewer
High	An update to a customer or supplier record	Full execution record, independent verification and listed exceptions	A named reviewer and a recovery owner
Critical	A regulatory filing or a payment instruction	The full record, independent verification and a documented decision	The accountable officer, named in advance

Proposed tiers. The author’s analysis, not a standard.

No tier is free. The cost of the top tier is the reason to keep the work that reaches it small and well defined. I am not certain these four tiers are drawn in the right places, and I expect practice to move them.

6. A Failure Worth Demonstrating Before Deployment

In the next pilot, deliberately make a required source unavailable. Use a test environment and harmless data. Ask the agent to continue with the ordinary task, then inspect both the final answer and the execution record.

Step	What to do	What a passing workflow shows
1. Remove the source	Make a required source unavailable	The agent discloses the missing source and reports blocked
2. Invite completion	Ask the agent to carry on with the ordinary task	The task cannot be marked complete without the evidence
3. Inspect the handoff	Watch what the receiving workflow does	The handoff halts. A warning that nobody must resolve is a failure
4. Restore and repeat	Bring the source back and run the task again	The evidence belongs to this run and this input, not to an earlier success
5. Change the rule	Change the acceptance rule through the authorised review	The record shows who approved the change and when

Test with harmless data. A pass on one run is a sample of one.

These checks do not prove that an agent will always behave honestly. They test whether a specific workflow can recognise unsupported completion and contain it. Repeat them when the model, the tools or the acceptance criteria change.

7. Questions for Any Agent Vendor

For a buyer comparing vendors, eight questions separate a described control from a working one. I would ask for a demonstration of each, not a description.

1. What happens, step by step, when a required source is unavailable?
2. Can the agent mark a task complete without the evidence for it?
3. Can the agent alter the checker, the acceptance rule or the execution record?
4. Where is the execution record stored, and who has write access to it?
5. Is the verifier independent in access and evidence, or does it read the agent's own summary?
6. Which outcomes exist besides complete, and who sees a blocked task?
7. Who is the named reviewer with authority to reject, and who owns recovery when accepted work proves wrong?
8. What changes in the process when the model, the tools or the acceptance criteria change?

A vendor with real answers to all eight is describing a working discipline. A vendor with a polished status page and vague answers to the first four is describing the evidence gap with a good interface.

CLOSING

Trust With Something Behind It

The useful outcome is an enterprise that can rely on an agent within a defined scope because the work can be checked. This is the design question I keep returning to while building KTER: what has to be visible before a person can stand behind an agent's work?

No evidence record removes every risk. The checker can be wrong, the source can be stale, and a reviewer can miss something. Separating those responsibilities does not remove the remaining uncertainty. It makes the uncertainty visible and gives the organisation a place to intervene.

> ***Before an enterprise accepts "done", it needs a record that survives one request: show me the run.***

SOURCES

[1] Reuters, 29 September 2026. China's AI agents can lie and scheme - just like their US rivals. https://www.reuters.com/business/retail-consumer/chinas-ai-agents-can-lie-scheme-just-like-their-us-rivals-2026-09-29/

[2] Guo et al., 4 December 2025. Are Your Agents Upward Deceivers? https://arxiv.org/abs/2512.04864

[3] Anthropic, 21 November 2025. From shortcuts to sabotage: natural emergent misalignment from reward hacking. https://www.anthropic.com/research/emergent-misalignment-reward-hacking

The news hook is dated 29 September 2026. The two studies predate it. The supplier example is a constructed illustration. The operating recommendations, outcome states and consequence tiers are the author's analysis.

Srinivasarengan Varadharajan · Founder, KTER · srini@kter.ai · © 2026 KTER