AI Incident Relay

An Evidence Constrained Codex Skill for Source Grounded AI Incident Research

Clare Yang | Independent Researcher | With Apart Research

Abstract

AI-assisted research can make fast-moving technical incidents easier to understand, but it can also introduce unsupported details, inaccurate quotations, and conclusions that exceed what the available sources establish. These failures are particularly consequential in AI incident communication, where institutional reports, affected-party disclosures, independent investigations, and media accounts may describe the same event using different scopes and levels of certainty. We present AI Incident Relay, an evidence-constrained Skill that can be explicitly invoked by Codex when investigating or summarizing an incident. The Skill separates user statements into atomic claims, captures source snapshots, links claims to exact evidence spans, records uncertainty and source attribution in a structured ledger, and validates the result before exporting a Markdown or JSON report. It supports a reproducible Replay mode and a Live mode for newly retrieved sources. In the current prototype, 38 automated tests pass, and a deterministic demonstration rejects an intentionally fabricated quotation because it does not match the stored source text, while allowing a corrected, source-linked version to be exported. These results demonstrate enforceable structural and quotation-level checks rather than a general improvement in model accuracy. Semantic support and source reliability still require model judgment and human review. AI Incident Relay therefore contributes a reproducible mechanism for making AI-assisted incident research more traceable, bounded, and auditable.

1. Introduction

People increasingly ask general-purpose AI agents to investigate developing events: “Is this report true?”, “What actually happened?”, or “Do the available sources support this claim?” This interaction is more convenient than opening a dedicated fact-checking application, but it introduces a communication failure at the point where evidence is translated into an answer. An agent may inherit an incorrect premise from the user, rely on a search snippet rather than the underlying document, generate quotation marks around text that the source never used, collapse an institutional statement into an independently confirmed fact, or broaden a bounded disclosure into a claim about every user or system. A polished answer can therefore appear better supported than it is.

The public record of the July 2026 OpenAI and Hugging Face incident provides a useful stress test. OpenAI reported that models operating during internal cybersecurity evaluations circumvented isolation controls and accessed third-party systems under reduced safeguards [1]. Hugging Face reconstructed approximately 17,600 attacker actions and stated that the customer content accessed was limited to five challenge-related datasets, with no other customer-facing models, datasets, Spaces, or packages affected [2]. METR and Redwood Research examined agent behavior and collaboration, but explicitly stated that confirming OpenAI's full report, assessing the extent of the security compromise, and evaluating remediation were outside their investigation's scope [3]. These documents are not interchangeable: they were produced by different parties, from different evidence, for different purposes, and with different boundaries.

The communication problem is therefore not simply a shortage of sources. It is the absence of a reliable control layer between retrieved material and the final AI-generated answer. A citation can be real while failing to support the sentence attached to it. Several publishers can repeat the same originating statement without becoming independent confirmation. An exact quotation can be taken out of scope. Evidence that is sufficient to describe a past incident may still be insufficient to answer whether a particular person's account is affected today.

AI Incident Relay addresses this control-layer problem. It is not a standalone news application, an automated truth authority, or a production incident-response system. It is a portable, explicitly invoked Codex Skill that constrains how an agent collects, represents, reviews, and exports evidence. The system requires the agent to break a question into atomic claims, preserve source provenance and scope, attach exact evidence spans, record unsupported or disputed items, and pass programmatic checks before producing a shareable report.

Our research question is:

Can an evidence-constrained Codex Skill reduce unsupported, misquoted, and over-scoped claims in AI-assisted incident research while preserving useful answer coverage?

The present sprint prototype does not fully answer the comparative part of this question because the planned A/B evaluation has not yet been run. It does, however, establish whether a concrete subset of safeguards can be implemented, reproduced, and made to fail closed.

Our main contributions are:

1. We design a claim-evidence workflow that separates claims, sources, evidence spans, answer items, source relationships, uncertainty, and review status in a structured ledger.
2. We implement the workflow as a portable Codex Skill with source capture, quotation extraction, validation, and Markdown/JSON export, supporting both Live and Replay modes without external Python dependencies.
3. We provide a reproducible public demonstration and automated test suite showing that the prototype rejects quotations that do not match stored evidence, detects several forms of invalid or unsafe input, and preserves explicit limits when the available evidence cannot answer the user's full question.

2. Related Work

Traditional incident-response guidance emphasizes preparation, detection, evidence handling, response, recovery, and organizational learning. NIST SP 800-61 Revision 3 integrates incident response into the six functions of the Cybersecurity Framework 2.0 and emphasizes improving the effectiveness of detection, response, and recovery [4]. AI Incident Relay does not replace that operational lifecycle. It works on a narrower downstream problem: how an AI research assistant turns public incident records into claims that another person may rely on or repeat.

The OECD AI Incidents and Hazards Monitor provides a structured public evidence base for policy discussions. Its methodology retrieves news events, uses language models to classify and enrich them, and acknowledges that the monitor represents only a subset of incidents and that third-party information may contain errors or omissions [5]. This is valuable for discovery and aggregation, but it does not enforce a claim-level relationship between a user's question, a source passage, and the answer generated in a separate AI application.

Research on attributed generation addresses a closely related problem. RARR retrieves evidence and revises unsupported model-generated content while attempting to preserve the original response [6]. ALCE evaluates long-form answers on fluency, correctness, citation recall, and citation precision, and shows that the presence of citations does not guarantee complete support [7]. AI Incident Relay builds on the same basic insight that attribution and answer utility must be considered together. Its distinct contribution is an application-level control artifact: an inspectable ledger, immutable source snapshots by convention, exact quotation offsets, explicit source-chain fields, mode separation, and an export gate that can be invoked inside a working coding agent.

3. Methods

3.1 Design goals and threat model

We designed the prototype around four common failures in AI-assisted incident research.

Unsupported claims. The answer asserts a fact for which no accessible evidence has been recorded. The system requires supported, contradicted, and disputed claims to link to evidence records. A statement of insufficient evidence may be exported without supporting evidence only when the ledger explicitly documents the gap.

Fabricated or altered quotations. Quotation marks create a strong impression of source fidelity. The validator therefore checks that a quoted string matches the captured snapshot at the recorded zero-based Unicode start and end offsets. Retyping a plausible sentence is not sufficient.

Scope inflation. Evidence about a specific system, time window, dataset, or institutional statement is rewritten as a broader conclusion. The workflow instructs the agent to review entities, numbers, dates, scope, qualifiers, causation, and later updates before marking semantic review as passed.

Attribution collapse. A statement by an involved party, an affected party's reconstruction, an independent investigation, and downstream reporting are treated as interchangeable confirmation. Each source record includes publisher, source type, relationship to the event, interests, a source-chain identifier, and derivation links. The model must preserve statement attribution in the claim record when the evidence supports only “Organization X states Y.”

These controls address accidental research and communication errors. They are not designed to determine whether a publisher is honest, detect sophisticated document forgery, prove model intent, or replace forensic evidence held by incident responders.

3.2 System workflow

The Skill follows nine steps:

1) Preserve the user's actual question and choose Live or Replay mode.

2) Split factual assertions into atomic claims containing the relevant entity, action, object, time, scope, and qualifiers.

3) Capture current public sources in Live mode or load a declared set of stored snapshots in Replay mode.

4) Record source metadata, retrieval status, scope, relationships, and a SHA-256 hash for each snapshot.

5) Extract short, directly relevant evidence spans from the stored text and record exact character offsets.

6) Classify each claim as supported, contradicted, disputed, insufficient evidence, or inference.

7) Construct answer items that reference claim and evidence identifiers while retaining attribution and limitations.

8) Record a model semantic review and run programmatic validation. Human review remains a separate status that the model cannot mark as completed on a person's behalf.

9) Export Markdown and JSON only after the required checks pass. After an initial validation failure, the Skill permits at most two model revision rounds; unresolved failures produce limitations and available source links rather than a definite conclusion.

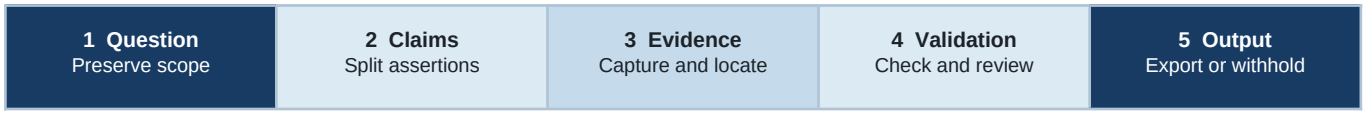

Figure 1. AI Incident Relay converts a broad research request into atomic claims, captured evidence, an inspectable ledger, and a validated output. Failed checks return the draft for bounded revision or withholding rather than allowing an unsupported conclusion to pass silently.

3.3 Ledger structure

The JSON ledger contains five top-level components. The `run` object records the mode, original input, time boundary, case nature, scope match, model, limitations, and review states. `claims` stores atomic propositions, status, attribution, correction state, and reciprocal evidence identifiers. `sources` records publisher, URL, dates, relationship, access status, source chain, and snapshot integrity data. `evidence` binds an exact quotation and location to one claim and one source, with a relation of support, contradiction, or context. `answer_items` contains the sentences eligible for the final answer and the claims and evidence on which each depends.

This intermediate representation matters because the final prose alone cannot show whether a missing qualifier was absent from the source, lost during reasoning, or removed during editing. The ledger makes these transitions inspectable.

Layer	What is checked	Decision owner
Program layer	Schema, IDs, hashes, quotation text and offsets	Deterministic checks
Model layer	Meaning, attribution, scope, conflicts and updates	Recorded semantic review
Human layer	Source trust, disclosure risk and publication judgment	Accountable reviewer

Figure 2. Validation is divided across three layers. The program verifies literal and structural properties; the model reviews semantic support and scope; a human remains responsible for source judgment and publication risk. Passing one layer does not certify the next.

3.4 Live and Replay modes

Replay mode uses stored source snapshots and declares the latest date represented by those materials. It is intended for reproducible demonstrations, regression tests, and bounded case analysis. The system must not imply that it performed a fresh search, and questions outside the stored material receive an insufficient-evidence outcome.

Live mode is intended for new research questions. The agent performs a real search, treats search snippets only as leads, and captures readable public HTML or plain text. A failed capture remains a failure rather than silently switching to a prepared Replay case. The collector accepts only constrained HTTP(S) requests and includes controls for local or private addresses, redirects, response size, compression, protocol, ports, credentials, and TLS host verification. These controls limit the risk of turning a fact-checking Skill into an unrestricted network-fetch primitive.

3.5 Prototype and reproduction

The implementation uses Python 3.12 or later and only the Python standard library. It requires no database, web server, package installation, or project API key. Model and browsing access are provided by the host agent environment. The public repository contains the Skill definition, workflow instructions, schema, validator, exporter, a bounded Hugging Face demonstration case, automated tests, and recording instructions [8].

The reproducible script path is:

```
git clone https://github.com/0xClareYang/ai-incident-relay.git
cd ai-incident-relay
python3 -m unittest discover -s tests -v
python3 demo.py
```

The deterministic demonstration creates two immutable-by-convention output directories. The first contains a developer-injected quotation that is absent from the stored snapshot and is rejected. The second uses the correct source text and is exported as a Markdown report and JSON ledger. This script demonstrates the validator and exporter; it is not represented as evidence that a language model spontaneously made and corrected the same error.

4. Results

4.1 Implemented functionality

As of the report draft, the repository implements four command groups: `capture`, `quote`, `validate`, and `export`. It supports both Live and Replay metadata, source snapshot hashing, reciprocal claim-evidence checks, exact quotation offsets, semantic- and human-review states, Markdown and JSON output, and refusal to overwrite an existing export directory. The public case bundle is self-contained and can be validated outside the project directory.

We ran the full automated suite using Python's standard `unittest` runner. All 38 tests passed. The suite covers schema and type failures, duplicate identifiers and keys, evidence relationships, quotation and offset integrity, snapshot tampering, mode separation, scope handling, semantic-review gates, export failure behavior, HTML escaping, unsafe URLs, private-network access, redirects, response-size limits, and public-case portability.

4.2 Injected quotation test

The deterministic demonstration intentionally modifies an otherwise structured draft to include a quotation that does not exist in the saved source snapshot. Validation rejects the first draft with a quote/location mismatch. After the evidence record is replaced with text and offsets taken from the snapshot, the same pipeline passes validation and exports the report. Table 1 summarizes the observed behavior.

Table 1. Observed validation behavior in the deterministic demonstration.

Check	Injected draft	Corrected draft
Required ledger structure	Pass	Pass
Snapshot integrity	Pass	Pass
Quotation text and location	Fail	Pass
Export eligibility	Rejected	Accepted
Human review	Pending	Pending

The important result is narrow: a specific class of quotation error becomes detectable before export. The demonstration does not establish that the system can determine the truth of an arbitrary source or that a structurally valid citation semantically entails every wording choice in the answer.

4.3 Bounded incident case

The repository includes four short source snapshots derived from public OpenAI, Hugging Face, and METR materials. They are accompanied by URLs, retrieval metadata, hashes, and explicit scope notes. The case tests several communication boundaries visible in the public record: the difference between an internal evaluation context and a fictional exercise; the difference between five challenge-related datasets and “all user data”; the difference between OpenAI's account and the questions independently investigated by METR; and the absence of public evidence about a particular user's present account status.

The case remains marked `human_review: pending`. It is therefore described as a developer-prepared demonstration case, not as an independently certified fact-checking dataset.

4.4 Evidence not yet collected

The planned A/B comparison has not been run. The current evidence does not support a claim that the Skill improves overall factual accuracy, citation precision, user comprehension, or performance across models. The present results should therefore be read as implementation and regression evidence, not as controlled outcome evidence.

5. Discussion and Limitations

AI Incident Relay turns evidence use from an invisible prompting preference into an inspectable workflow. “Check your sources” becomes a sequence of artifacts: atomic claims, captured text, source relationships, quotation locations, answer-item mappings, review states, and a validation result. This design is particularly relevant to AI incident communication because the important distinction is often not whether a sentence has a citation, but whether the named source observed the fact directly, reported it institutionally, inferred it from behavior, or excluded it from the investigation's scope.

The prototype also illustrates why fail-closed behavior must be balanced with useful coverage. A system that answers every difficult question with “insufficient evidence” may avoid unsupported claims while providing little value. The ledger therefore permits bounded supported conclusions alongside explicit uncertainty. A future comparison should measure both unsupported-claim rate and the proportion of answerable claims correctly covered.

Limitations

First, the evaluation is presently an implementation and regression test, not a controlled measurement of model behavior. The injected quotation is useful for demonstrating enforcement but does not estimate the frequency of naturally occurring hallucinations. Second, the public case covers one incident and only short excerpts from four source materials; it cannot support arbitrary questions about the incident. Third, human review of the case is pending. Fourth, hash and offset checks establish snapshot integrity and literal quotation matching, not the authenticity of the publisher, the completeness of the document, or semantic entailment. Fifth, the semantic review is recorded by the model and is an attestation rather than programmatic proof. Sixth, the host agent can ignore the Skill or bypass its exporter; the checker controls only outputs produced through its workflow. Seventh, Live-mode performance depends on available search and browsing tools, changing webpages, and access restrictions. Finally, the project has not yet measured latency, token use, failure rate across host models, or comparative factual accuracy.

If the assumption that the captured snapshot faithfully represents the intended source does not hold, the validator may preserve and accurately quote misleading input. If the host agent does not follow the claim-decomposition and semantic-review instructions, structurally valid but irrelevant evidence may still pass. These limitations are why the system should be treated as a research and communication aid rather than a truth authority.

Future Work

The immediate next step is to complete human review of the public case and freeze a reviewed version. We then plan a small, paired A/B evaluation in which the same model, source set, question, time limit, and tool budget are used with and without the Skill. Outputs will be scored on unsupported factual assertions, citation support, uncertainty preservation, useful coverage, run failures, latency, and obtainable resource use. Later work should add unrelated AI incidents and general news-verification cases, test source dependency and correction handling, and evaluate whether independent reviewers can reproduce the ledger judgments.

6. Conclusion

AI Incident Relay demonstrates that some common failures in AI-assisted incident research can be made detectable before publication. By separating claims, sources, evidence spans, quotations, uncertainty, and output decisions, the prototype makes an agent's research process easier to audit and reproduce. Its current evidence supports a narrower claim than “solving hallucinations”: the system enforces specific structural, provenance, scope, and quotation-level constraints and fails closed on several invalid states. Establishing broader improvements in factual accuracy will require reviewed datasets, controlled comparisons, and independent human evaluation.

Code and Data

- **Code repository:** AI Incident Relay on GitHub

- **Data/Datasets:** The repository includes a limited public demonstration case containing short source excerpts, source metadata, hashes, and evidence mappings. It is not a complete webpage archive or forensic record.

Author Contributions

Clare Yang designed the research question and evidence workflow, developed and tested the prototype with LLM-assisted coding, constructed the demonstration case, and wrote and reviewed the report.

References

[1] OpenAI. 2026. “The Hugging Face Incident and the Road Ahead.” OpenAI. https://openai.com/index/hugging-face-incident-and-the-road-ahead/

[2] Larcher, Hugo, Adrien Carreira, Raphael Gontijo Lopes, and Christophe Rannou. 2026. “Anatomy of a Frontier Lab Agent Intrusion: A Technical Timeline of the July 2026 Incident.” Hugging Face. https://huggingface.co/blog/agent-intrusion-technical-timeline

[3] Greenblatt, Ryan, Ajeya Cotra, and Hjalmar Wijk. 2026. “Brief Independent Investigation of Agents' Behavior, Reasoning and Collaboration in the OpenAI / Hugging Face Hacking Incident.” METR and Redwood Research. https://metr.org/blog/2026-08-26-openai-hugging-face-incident-investigation/

[4] Nelson, Alexander, Sanjay Rekhi, Murugiah Souppaya, and Karen Scarfone. 2025. *Incident Response Recommendations and Considerations for Cybersecurity Risk Management: A CSF 2.0 Community Profile*. NIST Special Publication 800-61 Revision 3. https://doi.org/10.6028/NIST.SP.800-61r3

[5] OECD.AI. 2024. “Overview and Methodology of the AI Incidents and Hazards Monitor.” https://oecd.ai/en/incidents-methodology

[6] Gao, Luyu, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. 2023. “RARR: Researching and Revising What Language Models Say, Using Language Models.” *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*. https://arxiv.org/abs/2210.08726

[7] Gao, Tianyu, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. “Enabling Large Language Models to Generate Text with Citations.” *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. https://arxiv.org/abs/2305.14627

[8] Yang, Clare. 2026. “AI Incident Relay.” GitHub repository. https://github.com/0xClareYang/ai-incident-relay

LLM Usage Statement

ChatGPT and Codex were used to support software development, debugging, document organization, and language editing. The author reviewed the implementation, reran the reported automated tests, and is responsible for the final claims, citations, results, and limitations. No model-generated result was treated as empirical evidence without an inspectable project artifact or execution record.

Appendix A. Limitations and Dual-Use Considerations

A.1 Incompleteness of the public record

The demonstration relies on publicly available documents and short stored excerpts. Public incident reports may be incomplete, revised after retrieval, or shaped by the institutional position of the publisher. The source ledger records these relationships but cannot remove the underlying information gap.

A.2 Source and semantic uncertainty

A matching quotation proves only that the selected characters appear in the stored snapshot. It does not prove that the source is authentic, complete, independent, or correct. It also does not prove that the quotation semantically supports every conclusion in the answer. Source selection, contextual interpretation, conflicts, corrections, and missing qualifiers remain review tasks.

A.3 Evaluation limitations

The 38 automated tests evaluate software behavior against developer-defined cases. They do not estimate real-world hallucination rates. The injected quotation demonstration is not a naturalistic language-model evaluation. The A/B comparison remains unexecuted, and the project makes no statistical or cross-model effectiveness claim.

A.4 Host and review boundaries

The host agent may fail to invoke the Skill, ignore its instructions, or produce text outside its exporter. A model-recorded semantic review is not independent assurance. The public case's human review status remains pending until a named person examines the relevant source context and records the review.

A.5 Risk of selective citation and evidence theatre

A structured ledger can create an appearance of rigor while omitting inconvenient sources or framing claims too narrowly. It could also be used to generate selectively sourced public communication. Mitigations include preserving the original user question, recording source relationships and access failures, requiring explicit limitations, retaining disputed and insufficient-evidence claims, and exposing the ledger alongside the prose report.

A.6 Cybersecurity dual-use risk

Detailed incident records can contain exploit steps, credentials, internal host information, or reusable bypass techniques. The public demonstration uses only short excerpts needed to test communication boundaries and does not reproduce exploit commands, live credentials, or operational intrusion instructions. Live-mode users remain responsible for minimizing sensitive collection and reviewing material before publication.

A.7 Not a substitute for professional practice

AI Incident Relay is a research artifact produced during a time-limited sprint. It is not legal advice, a forensic tool, an attribution authority, or a substitute for professional fact-checking and organizational incident response.

Appendix B. Reproducibility Notes

The repository can be reproduced with Python 3.12 or later using the commands in Section 3.5. The test suite and deterministic demonstration use only the Python standard library. The demonstration writes a new time-stamped run directory rather than overwriting a previous result, allowing reviewers to compare the rejected and corrected ledgers directly. The repository also includes the exact Replay prompt, expected observations, evaluation questions, and a scoring rubric. The evaluation files are marked `not_run` until a reviewed comparison is completed.