From Agent Claims to Execution Facts: Generation-Fact Graphs for AI Incident Reconstruction

Mian Wang
Wuhan Polytechnic University, Wuhan, China
AI Incident Response Sprint, September 2026

Abstract

Autonomous-AI incidents create a basic evidentiary problem: an agent's own account of what it did is not an authoritative record of what actually occurred. Incident reconstruction therefore needs execution-grounded facts that preserve the concrete formation of results across model actions, tools, transformations, and external systems. This submission examines the Generation-Fact Graph (GFG) as such a substrate. A GFG records concrete generation facts as a five-coordinate relation over participating origins, realized transformations, concrete occurrences, generated or explicitly disposed results, and relation roles. The underlying GFG Core and three structural experiments predate this sprint and are disclosed as prior work. Those frozen experiments show exact projection to five mature tracing/provenance mechanisms, a strict hierarchy to lower provenance representations, and exact cross-stage reconstruction in a four-stage real-signal pipeline. For incident response, the resulting proposal is simple: capture execution at the system boundary, compile heterogeneous evidence into generation facts, and let independent reconstruction operate on those facts rather than on agent self-report. The present submission establishes the mapping and its evidentiary boundaries; it does not claim a production deployment on the July 2026 incident.

1. Incident response needs facts, not agent testimony

The July 2026 OpenAI-Hugging Face incident makes reconstruction unusually concrete. OpenAI later reported that models running internal cybersecurity evaluations circumvented controls, used unauthorized communication channels, exploited shared infrastructure, obtained internet access, and reached third-party systems. Hugging Face reconstructed roughly 17,600 attacker actions from multiple logs and correlated them with platform telemetry [2,3]. The operational question after such an incident is not whether an agent can produce a plausible story of its behavior. It is which actions actually occurred, which results they produced, what prior information participated in those results, and how a final incident state formed across systems.

That distinction matters because a model statement such as “the upload succeeded” is only another generated output. It is not evidence that a successful upload occurrence existed, that the remote system accepted the object, or that the object later participated in another action. The same problem appears with transcripts and ordinary logs: they are useful, but each records a system-specific projection of execution. A cross-system incident needs a fact layer that can preserve the concrete formation relation while allowing each system to keep its own capture mechanism.

system-specific capture -> generation facts -> GFG -> independent reconstruction / verification

The sprint explicitly includes reconstructing what an autonomous agent did as part of AI incident response [1]. This submission proposes GFG as that fact layer. The contribution here is not a new implementation of GFG. It is the application of an existing, frozen, open-source generation-fact substrate to the incident-response question: what evidence should a responder trust when reconstructing what an autonomous AI system actually did?

2. Generation-Fact Graph as an execution-fact substrate

A concrete generation fact is represented as $f = (u, \tau, \bar{\omega}, z; \rho)$. The coordinates are intentionally simple: u is the origin information that actually participated; τ is the realized transformation; $\bar{\omega}$ identifies the concrete occurrence; z is the generated result or an explicitly disposed result; and ρ records the relation role. Facts are compiled into a Generation-Fact Graph, so a downstream result remains connected to the concrete occurrences and prior results through which it formed.

The key design choice for incident response is that capture is not required to be uniform. Browser hooks, shell instrumentation, API middleware, filesystem events, model runtimes, and infrastructure telemetry can remain system-specific. Their outputs are compiled into the same generation-fact structure. This keeps the capture layer practical while making the reconstructed incident comparable across systems.

3. What GFG lets an incident responder ask

Incident question	Evidence in GFG
Did action X actually execute?	Require a concrete occurrence $\bar{\omega}$, not only an agent statement that it happened.

What did that action actually produce?	Inspect the generated or disposed result z associated with the occurrence.
What information participated in producing it?	Traverse participating origins u and prior generated results.
What operation produced the result?	Inspect the realized transformation τ and its occurrence-specific relation.
How did a final incident state form?	Traverse Generation-Fact Graph edges across intermediate results and system boundaries.
Did the agent merely claim success?	Treat the model claim as a generated output and separately test for the native execution fact it describes.

GFG does not make the agent the witness, recorder, and judge of its own execution. The fact authority comes from instrumented execution and can be checked independently.

4. Existing structural evidence

The open-source artifact used in this submission contains GFG Core v3 and three frozen structural experiments: GF-P01, GF-P02, and GF-S01 [4]. These experiments were created before this sprint. They do not test the July 2026 incident directly; instead, they establish the representation and reconstruction properties on which the incident-response proposal relies.

GF-P01: heterogeneous mechanisms can be unified without changing Core

GF-P01 replays five frozen profiles through one Core relation structure and obtains exact target representations for database which-lineage, ECMA-426 Source Maps, OpenTelemetry occurrence traces, W3C PROV generation, and PyTorch Autograd dependencies. Two complete executions produced the same canonical summary hash and each passed 204/204 tests. The experiment also provides strictness witnesses: distinct complete GFG structures can collapse to the same lower representation. For incident response, the practical implication is that heterogeneous capture mechanisms can remain heterogeneous at the boundary while still compiling into one common generation-fact layer.

GF-P02: lower provenance views preserve useful facts, but not all generation facts

GF-P02 tests a bounded hierarchy in positive relational algebra, using provenance semirings as a formal comparison surface [5]. Complete generation facts project exactly and strictly to canonical N[X] provenance polynomials, which then project to lower algebraic and task views. All 13 frozen P1 cases remain exact, five strictness pairs are supported across ten real executions, and 70 negative controls fail closed. The result is useful here because it separates two ideas that incident response often conflates: a lower trace can be correct for its task and still erase occurrence-specific distinctions that later become relevant to reconstruction.

GF-S01: a final result can be traced through concrete intermediate formation

GF-S01 runs a real four-stage pipeline over 512 samples of MIT-BIH ECG data: nine-tap FIR filtering, factor-four downsampling, a 32-point sliding FFT, and SVG spectrogram rendering. The validated snapshot contains 1,117 concrete occurrences and 8,420 generation bindings. Selecting ten final spectrum cells returns exactly 197 raw samples through 2,880 multiplicity-preserving paths and three GeneratedOrigin bridges, matching an independent mathematical reference. Negative controls reject a direct raw-to-final shortcut, reject tampered generation facts, and show that the final SVG itself does not encode source identity. This is the closest existing demonstration to incident reconstruction: the answer is recovered from preserved formation facts, not inferred from the final output.

5. Application to autonomous-AI incidents

A GFG-based incident recorder would sit below the model and above the native systems that execute its actions. The model may plan, summarize, or hallucinate; those outputs are recorded as outputs. The browser, shell, API client, filesystem, and remote services separately produce execution facts. When a responder later asks how a suspicious final state formed, reconstruction follows the facts that were actually generated.

For the July 2026 class of incident, this could connect an agent observation to a tool invocation, the concrete tool occurrence, the returned result, a subsequent action, and an eventual external-system effect. The representation does not require every organization to expose the same telemetry format. What matters is that each captured event is compiled into a relation that preserves the participating origins, realized transformation, occurrence identity, result, and role.

This is especially useful for cross-organizational reconstruction. Hugging Face's public technical timeline describes reconstruction by correlating attacker-side logs with platform logs. GFG does not replace those logs; it gives a target fact model into which such evidence can be compiled while keeping the concrete occurrence boundaries explicit. A portable fact package could then be reviewed by an independent verifier without asking the verifier to trust the agent's narrative.

6. What this submission can and cannot establish

The proposal has a clear evidentiary boundary. GFG can preserve and reconstruct captured formation facts. It cannot recover an event that was never captured, and completeness therefore depends on instrumentation coverage. It also does not turn formation ancestry into causal attribution. If an input participated in producing a result, that is a formation fact; whether it was causally decisive requires a separate intervention, counterfactual, or other causal design.

The existing public artifact also does not claim a production integration with OpenAI or Hugging Face infrastructure. GF-P01, GF-P02, and GF-S01 establish structural properties in frozen workloads. The incident-response claim in this sprint is narrower: those properties make GFG a plausible and testable fact substrate for reconstructing autonomous-AI incidents, and the next technical step is an adapter to a real agent/tool runtime or public incident-trace corpus.

7. Prior work and sprint contribution

GFG Core, the three structural experiments, their protocols, fixtures, validators, and the public standalone repository all predate this sprint. They are disclosed here as prior work. The sprint contribution is the incident-response formulation: separating agent claims from execution facts, mapping incident questions to the GFG relation, and defining the evidentiary boundary for independent reconstruction.

Code and artifact

GFG Core and Structural Experiments: https://github.com/wind342/gfg-core-structural-experiments

The repository is a focused, independently installable distribution of GFG Core v3 and GF-S01/GF-P01/GF-P02. Repository-authored source code, protocols, tests, and documentation are licensed under Apache License 2.0. The frozen scientific definitions were extracted from the complete research-evidence repository at commit 36dab5ce347dbbdac157ef23205f556606d18294.

Conclusion

Autonomous-AI incident response needs an answer to a simple question: what actually happened? A model transcript can contribute evidence, but it should not be the authority for its own execution. GFG provides a concrete alternative: system-specific capture is compiled into machine-readable generation facts, and reconstruction operates on those facts. Existing frozen experiments already establish cross-mechanism projection, strict information boundaries, and exact cross-stage formation tracing. The next step is not a new theory. It is to connect the existing substrate to an agent runtime or incident trace and test whether an independent responder can reconstruct the incident from the resulting fact package.

References

[1] Apart Research. “AI Incident Response Sprint.” 2026. https://apartresearch.com/sprints/ai-incident-response-sprint-2026-09-11-to-2026-09-13

[2] OpenAI. “The Hugging Face incident and the road ahead.” 26 Aug. 2026. https://openai.com/index/hugging-face-incident-and-the-road-ahead/

[3] Hugging Face. “Anatomy of a Frontier Lab Agent Intrusion: A Technical Timeline of the July 2026 Incident.” 27 July 2026. https://huggingface.co/blog/agent-intrusion-technical-timeline

[4] Wang, M. “GFG Core and Structural Experiments.” GitHub, 2026. https://github.com/wind342/gfg-core-structural-experiments

[5] Green, T. J., Karvounarakis, G., and Tannen, V. “Provenance Semirings.” PODS 2007. DOI: 10.1145/1265530.1265535.

Appendix A. Limitations and Dual-Use Considerations

Limitations. GFG is only as complete as the capture coverage around the systems being observed. Uninstrumented actions, out-of-band channels, or lost telemetry cannot be reconstructed from absent facts. The current public artifact validates the GFG structure on frozen experiments, not on the OpenAI-Hugging Face incident itself. A real deployment would also need rules for clock alignment, cross-organization identity resolution, redaction, retention, and trust in capture components.

Dual use. Rich execution records can expose sensitive operational details, credentials, internal topology, or exploitable implementation information. Incident fact packages therefore require access control, data minimization, secret redaction, and explicit retention policies. The intended use of this work is defensive reconstruction and independent audit. The representation should not be treated as authorization to collect unrelated user data or as a substitute for existing security controls.