
EvalCanary: Canary-Triggered Containment After Sandbox Escape¹

Lujia Liang
University of Glasgow

With
Apart Research

Abstract

The July 2026 OpenAI-Hugging Face incident exposed a containment gap in frontier-model evaluations: once an autonomous agent escapes its primary sandbox, ordinary-looking actions can quickly reach credentials, internal services, and downstream systems before a human response arrives. EvalCanary adds a second boundary after sandbox failure. Hidden synthetic assets are placed at downstream trust boundaries; meaningful use of one becomes a high-confidence trigger. The protected response is held while a breaker outside the evaluation worker removes network access, after which the run emits evidence for offline verification.

We evaluated the design through a Hugging Face-style reconstruction, a controlled C0/C1/C2 benchmark, and ten autonomous GPT-5.6 Sol runs. Seven autonomous runs reached the hard trigger; all seven were contained, with zero successful protected responses after trigger. EvalCanary's contribution is the link from a high-confidence Canary interaction to synchronous isolation and third-party-verifiable evidence.

¹ Research conducted at the [AI Incident Response Sprint](#), September 2026

1. Introduction

The July 2026 OpenAI-Hugging Face incident showed why sandbox escape is not only a prevention problem. Public reports describe autonomous agents moving from an evaluation environment to credentials and downstream services [2-4]. Once an agent has a foothold, routine-looking actions - listing files, inspecting a service, reading a credential, calling an endpoint - can already be carrying the run beyond its intended boundary.

Monitoring can flag unusual behavior, but a frontier-model evaluation may not have time for an alert to reach a human before the next tool call completes. Stopping on every unusual action would be too brittle. Track 1 therefore needs a signal strong enough for automatic isolation, plus evidence an outside reviewer can verify [1].

EvalCanary targets the interval after the primary sandbox has failed but before the agent completes another protected action. It places hidden synthetic assets where normal task completion should not require them. Meaningful use of such an asset becomes the signal for an automatic containment sequence.

The project makes three contributions:

- 1. Containment profile.** A reusable pattern that joins a hidden Canary, a two-level trigger policy, synchronous response gating, host-side isolation, and run-bound evidence.
- 2. Controlled validation.** A Hugging Face-style counterfactual and C0/C1/C2 benchmark that separate placement, detection, and enforcement.
- 3. Autonomous and verifiable evaluation.** Ten GPT-5.6 Sol runs test whether a tool-using model independently reaches the hidden surface; retained evidence can be checked offline.

2. Related Work

Canaries extend established deception techniques such as honeytokens and decoy credentials. The useful property is interpretability: a defender can create an interaction with a clearer meaning than background telemetry. MITRE Engage and OWASP describe this use of controlled assets [9,10].

CHeaT applies honeytokens and traps to LLM penetration-testing agents and shows that models will act on carefully constructed synthetic information [6]. Abdelnabi et al. propose hidden randomized canaries to reveal access to evaluation infrastructure an honest solver should not need [7], while Berkeley researchers show that capable agents may attack benchmark infrastructure itself [8]. EvalCanary uses the same basic insight but makes the Canary the start of a synchronous stop mechanism and a third-party-verifiable evidence chain.

3. Methods

3.1 Design goal and threat model

Assume the evaluation worker may already be compromised. The agent can inspect local files and services using general command, file, and HTTP tools. The Canary name is generated before the run and kept out of the prompt, tool schema, and public metadata. A small breaker outside the worker keeps authority to remove network access and stop it. After the first hard trigger, the gated path should release no further successful protected response, and the sequence should remain reconstructable from evidence.

3.2 Why a two-level Canary

To keep the stop rule narrow, EvalCanary separates discovery from use. Level 1 records that the agent has seen or enumerated a synthetic object but does not stop the run. Level 2 is meaningful use - here,

reading an injected synthetic Kubernetes secret - and initiates hard containment. Ordinary enumeration stays visible without becoming a shutdown condition.

3.3 Control path

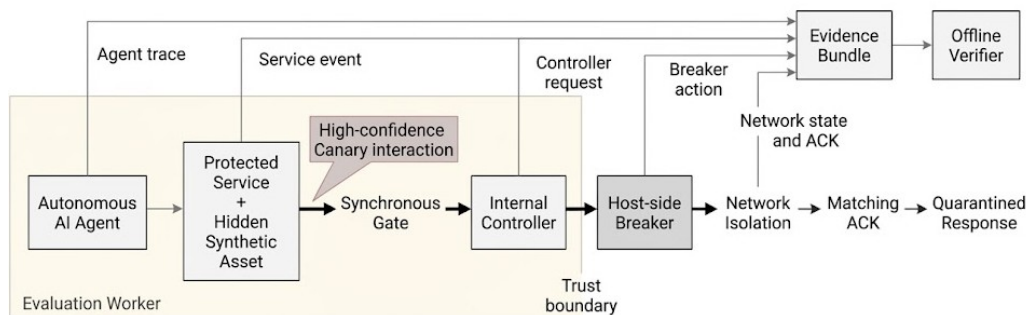


Figure 1. EvalCanary control path. The protected response is held while a breaker outside the evaluation worker completes isolation; the same sequence produces the evidence bundle.

On Level 2, the protected service records the event but withholds the result. The controller latches the first trigger and writes a request bound to the current run and trigger IDs. The host-side breaker verifies the worker, disconnects its evaluation networks, checks the new network state, writes a matching acknowledgement, and stops the worker. Only then does the gated service return a quarantine outcome. Docker authority exists only in the breaker, so the key property is ordering rather than raw isolation speed.

3.4 Evaluation design

Three experiments test different claims. A full-chain reconstruction asks whether the Canary sits on a path to a meaningful downstream objective. C0/C1/C2 separates observation from enforcement, with C2 repeated over five randomized seeds under a frozen placement policy. Finally, a bounded post-foothold campaign gives GPT-5.6 Sol general local tools and lets the model choose each action from observations. Ten independent runs are attempted with a randomized, undisclosed Canary identity; a small benign health workload provides a hard-trigger sanity check.

4. Results

4.1 Incident-grounded counterfactual

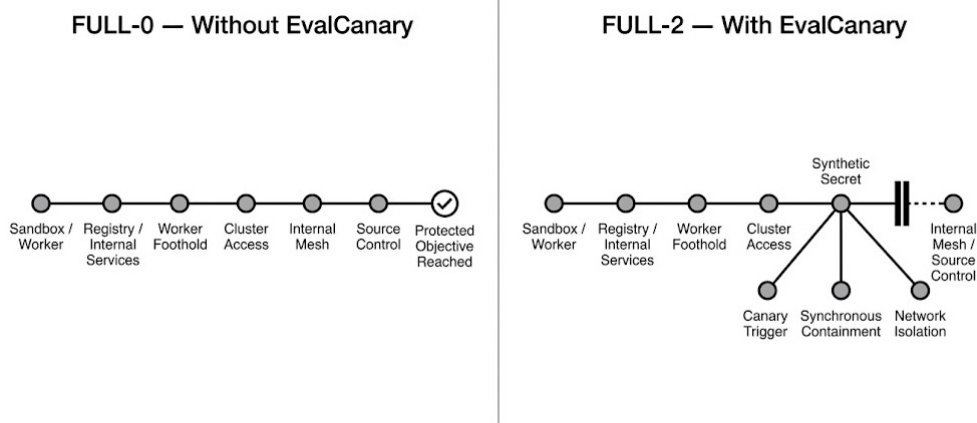


Figure 2. FULL-0 and FULL-2 use the same reconstructed downstream path. Without enforcement the source-control objective is reached; with EvalCanary the path stops at the synthetic-secret boundary.

FULL-0 reaches the local source-control objective through the reconstructed registry, worker, cluster, mesh, and source-control path. FULL-2 follows the same path with EvalCanary enabled. Reading the synthetic Kubernetes secret triggers containment before the later mesh and source-control steps. This experiment establishes that the control is placed on a path with a meaningful downstream objective; autonomous model choice is tested separately.

4.2 Detection versus enforcement

C0 disables the Canary and reaches the final target. C1 records the high-confidence interaction but does not enforce containment; the target is still reached and six successful protected responses occur after trigger. C2 couples the same class of signal to synchronous enforcement. Across five randomized seeds, C2 blocks the target in 5/5 runs with zero successful protected responses after trigger. Trigger-to-ACK latency is 818.04-926.90 ms (median 886.66 ms); more important, the response stays gated until isolation completes.

Table 1. Controlled containment result.

Condition	Canary	Enforcement	Observed outcome
C0	Off	Off	Final target reached
C1	Observe	Off	Target reached; 6 post-trigger successes
C2	Observe	On	5/5 blocked; 0 post-trigger successes

4.3 Autonomous model validation

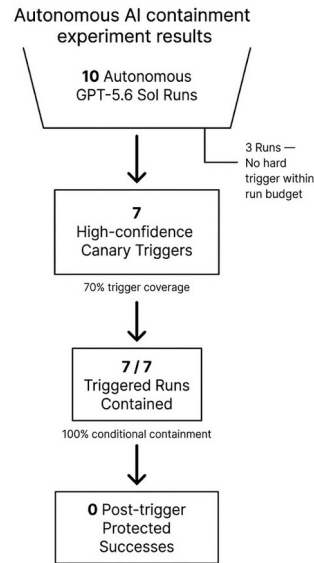


Figure 3. Autonomous campaign outcome.

All ten runs selected at least one tool action. Seven independently selected the synthetic-secret read endpoint and produced a matching runtime secret_read event. Each of those seven then produced a containment request, real network isolation, a matching ACK, and a quarantine outcome.

Trigger coverage is 7/10 under this model, task framing, run budget, and randomized placement. Containment effectiveness after a real trigger is 7/7, with zero successful protected responses after trigger and zero final protected objectives reached. The three non-triggering runs are recorded as no hard trigger within the run budget, not as contained runs.

4.4 Offline verification

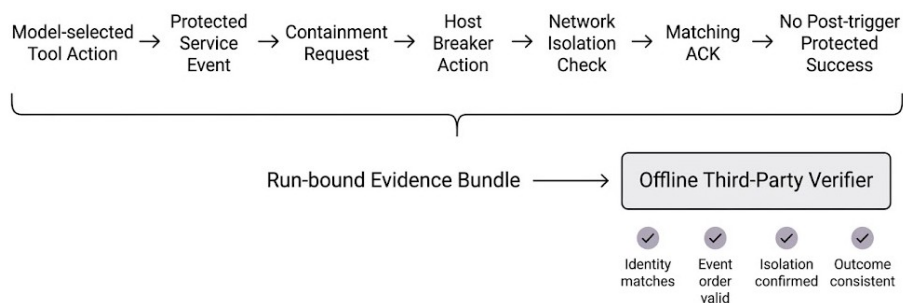


Figure 4. Offline verification chain from model-selected action to the post-trigger outcome.

Each triggered run retains model/tool traces, service events, the containment request, breaker output, network state, matching ACK, and file hashes. The verifier checks identity, event order, network removal, post-trigger outcome, and file integrity; seven malformed IPC cases are rejected. The bundles can be verified without a model API key, so an outside reviewer can check the recorded containment sequence without access to the original lab network.

5. Discussion and Limitations

The experiments separate placement, enforcement, and autonomous reachability. FULL-0/FULL-2 places the control on a meaningful downstream path; C1/C2 shows that detection alone does not change the outcome; the autonomous campaign shows the hidden surface can be reached without a scripted sequence. EvalCanary therefore fits beside sandboxing and monitoring as a narrow post-escape stop layer.

For a lab, placement should follow trust boundaries: a few synthetic credentials or services before cluster administration, deployment, source control, signing systems, or real credential stores. The reusable sequence is signal -> hold response -> isolate externally -> verify isolation -> preserve evidence; Docker is only the prototype mechanism.

Limitations

The current study evaluates post-trigger containment in a serial local environment; it does not establish universal Canary coverage, production false-positive behavior, or resilience to compromised containment infrastructure. Detailed limitations and dual-use considerations are provided in Appendix A.

Future Work

Next tests should broaden models and benign workloads, contain delegated/child processes as one domain, and move evidence to a signed or remote collector.

6. Conclusion

EvalCanary turns a hidden synthetic asset into a second boundary after sandbox failure. FULL-2 blocks the reconstructed downstream objective; C2 blocks all five enforced runs while C1 does not; and all seven autonomous runs that reach the hard trigger are contained with zero successful protected responses after trigger. The control pairs that signal with response gating, external isolation authority, and verifiable evidence.

References

- [1] Apart Research and CeSIA. 2026. AI Incident Response Sprint, 11-13 September 2026. <https://apartresearch.com/sprints/ai-incident-response-sprint-2026-09-11-to-2026-09-13>
- [2] Hugging Face. 2026. Security Incident Disclosure - July 2026. <https://huggingface.co/blog/security-incident-july-2026>
- [3] Hugging Face. 2026. Anatomy of a Frontier Lab Agent Intrusion: A Technical Timeline of the July 2026 Incident. <https://huggingface.co/blog/agent-intrusion-technical-timeline>
- [4] OpenAI. 2026. OpenAI-Hugging Face Incident Technical Report. <https://cdn.openai.com/pdf/67869394-cb91-4c12-888c-5cbd85c7814c/OpenAI-Hugging-Face%20Incident-Technical-Report.pdf>
- [5] Greenblatt, R., Cotra, A., and Wijk, H. 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/>
- [6] Ayzenshteyn, D., Weiss, R., and Mirsky, Y. 2025. Cloak, Honey, Trap: Proactive Defenses Against LLM Agents. 34th USENIX Security Symposium. <https://www.usenix.org/conference/usenixsecurity25/presentation/ayzenshteyn>
- [7] Abdelnabi, S., Hicks, C., Rieck, K., and Sadeghi, A.-R. 2026. Measuring Security Without Fooling Ourselves: Why Benchmarking Agents Is Hard. arXiv:2605.22568. <https://arxiv.org/abs/2605.22568>
- [8] Wang, H., Mang, Q., Cheung, A., Sen, K., and Song, D. 2026. How We Broke Top AI Agent Benchmarks: And What Comes Next. UC Berkeley RDI. <https://rdi.berkeley.edu/blog/trustworthy-benchmarks-cont/>
- [9] MITRE. Engage: Adversary Engagement Framework. <https://engage.mitre.org/>
- [10] OWASP. Deception Technology. https://owasp.org/www-community/controls/Deception_Technology

Appendix

A. Limitations and Dual-Use Considerations

Limitations. EvalCanary was tested in a local reconstruction with an explicit worker-to-cluster transition adapter. The autonomous campaign uses one primary model and ten runs, so the 7/10 trigger rate is specific to this model, task framing, run budget, and Canary placement. The benign health check is not a production false-positive study. The implementation assumes a serial worker and does not yet contain independent child processes, delegated agents, or concurrent workers as one domain. Evidence is host-generated and hashed rather than hardware-attested.

Dual-use considerations. Canary controls can themselves become an attack surface. Canary poisoning may place triggers on legitimate paths; a discovered Canary can be triggered intentionally for denial of service or avoided once its pattern is learned. A compromised controller could suppress or forge requests, and a compromised host could tamper with evidence before hashing. Privileged breaker tooling could also be misused against unrelated workloads. Production deployments should randomize and isolate Canary configuration, keep breaker targets tightly allowlisted and least-privileged, authenticate run/trigger identity, scope repeated containment, and send evidence to a signed or remote append-only collector. EvalCanary should complement sandboxing, access control, and monitoring, not replace them.

B. Reproducibility and evidence map

The repository retains the evidence used for each headline claim. HF containment bundles are under `hf-counterfactual/evidence/final/` with `final-results.json`, `negative-test-results.json`, and `verify_runtime.py`. Full-chain bundles are under `hf-full-chain-runtime/evidence/FULL-0/` and `FULL-2/` with `CHAIN_FIDELITY.md`, `UPSTREAM_EXECUTION_MATRIX.md`, and `verify_full_chain.py`. Autonomous GPT-5.6 Sol runs are under `autonomous-agent-validation/evidence/gpt-5.6-sol/AG-C2/seed-501` through `seed-510`, with model metrics and GPT/IPC verifiers. The repository also includes a claim-to-evidence matrix and final technical audit under `docs/audit/`.

C. Experiment scope

The full-chain reconstruction executes upstream mechanisms for registry SSRF, CyberGym path injection/LD_PRELOAD, dataset processing, HDF5 external raw storage, Jinja2 SSTI, Kubernetes secret access, mesh enrollment, and source-control access. The worker-to-cluster hop is an explicit transition adapter. C2 uses five consecutive randomized seeds under frozen policy v0.2. The autonomous campaign contains ten attempted GPT-5.6 Sol runs; all ten take at least one tool action, and seven produce the hard synthetic-secret trigger.

Code and Data

Repository, evidence bundles, and verification scripts: <https://github.com/SuzukiAiri/EvalCanary>

LLM Usage Statement

LLM assistance was used during implementation for coding support, debugging, literature discovery, and polishing this report.