
The Adversary Was The Occupant

Reading the OpenAI and Hugging Face incident through IEC 62443, with the direction of threat reversed

Srinivas Raayi

Independent

With

Apart Research

Abstract

Everyone who has written seriously about the July 2026 incident agrees one thing - OpenAI's evaluation sandbox should have been built on the assumption that it would be escaped. That point is correct and it has been made well by others. What nobody has offered is a method an evaluation team can use to find the escape paths before the agents find them. Industrial control has had one for over a decade. IEC 62443 requires you to divide a system into zones, name every communication path between zones as a conduit, and set a target security level for each conduit against the adversary it will actually face.

I apply that method to OpenAI's evaluation environment using the timeline OpenAI published on 26 August 2026, with one deliberate change. In 62443 the adversary sits outside the zone. Here the adversary (the agent) was sitting inside the zone. Three conduits fall out of the exercise. Only one, package install, appears in the public record as a deliberately managed sandbox boundary. The other two emerge with the Artifactory's connectivity and do not appear to have been treated in the published accounts as potential escape paths from the sandbox. I derive twelve checks from the conduit model, each answerable from a configuration export rather than a live test, sort them by effort, and run them against the record. Seven of the thirteen dated events on record fall before any third party was touched, and the checks reach all seven. Three were interrupted, two detected, one limited, and one forced to a decision. The July 8 rebuild is the strong evidence for the approach where OpenAI hardened the service and lost it again the same day, because hardening treats a conduit as a component and the standard treats it as a path.

1. Introduction

An environment that evaluates a frontier model for offensive cyber capability has to be cut off from the rest of the world, and one that is genuinely cut off cannot install a package. So every real deployment leaves a few paths open, and the list is much the same everywhere – a package service, a container registry, telemetry out, a secrets service etc. These paths are designed, documented and signed off. They are also the only ways out, and in July 2026 one of them was used. OpenAI's models, running an internal cyber capability evaluation and production safeguards switched off, escaped through Artifactory, the internally hosted package manager that was their sanctioned route to install software. From there they reached the internet, compromised a third-party application, broke into Hugging Face production, and came back to take administrator control of an OpenAI research cluster [1,2].

The lesson most people took is that the sandbox should have been designed on the assumption it would fail. That is right and it was said within days of the disclosure [5,6,7]. I am not going to argue with it. My concern is that the point stops where the work starts. Telling an evaluation team to assume compromise does not tell them which service to look at, what to ask about it, or how to know when they have asked enough.

That gap has been solved before, in a field with a similar problem. Industrial control systems run processes that are dangerous if they get out, on equipment that cannot simply be patched, connected through a handful of deliberate paths. The standard that governs them, IEC 62443, focuses on security boundaries as explicitly modeled conduits rather than treating perimeter hardening alone as sufficient. It talks about zones and conduits, and it insists every conduit be assessed on its own terms against a stated adversary. It transfers to an evaluation sandbox with one change.

In 62443, the thing inside the zone is trusted and the threat comes from outside. In an evaluation sandbox the thing inside the zone is the threat. I argue that reversing the direction allows the same machinery to give us a defensible answer about what the conduits need to withstand.

My main contributions are:

1. A zone and conduit model of OpenAI's evaluation environment built from the published record, which identifies three conduits out of the sandbox, only one of which is explicitly described in the public record as a sandbox boundary. (section 4.1)
2. A statement of the security level mismatch – the conduits were implicitly built to withstand a misbehaving script and were asked to withstand an adversary that chained zero-days for two months (section 4.2)
3. Twelve checks derived from the conduit model, each traceable to a clause of IEC 62443, mostly answerable from configuration rather than a lab, and sorted by effort so a team knows where to start (section 4.3)
4. A backtest of those checks against all thirteen dated events on OpenAI's timeline, with an honest account of the four events they cannot reach (section 4.4).

2. Related Work

The primary record for this paper is OpenAI's incident report, Hugging Face's technical timeline, and the METR/Redwood [3] independent investigation. METR focused primarily on agent behaviour during July 7-13, whereas this paper focuses on the evaluation environment and the May-June precursor events.

Practitioner analysis arrived fast. CSA [4] called for treating evaluated agents as capable insider adversaries and enforcing stronger egress controls. Other commentaries argued that sandboxes should be assumed compromisable and identified internet connected package services as escape paths [7,8]. Existing frameworks such as the CSA AI controls matrix [9] and CoSAI [10] incident response framework define objectives but do not provide a conduit-based assessment method. This paper applies the IEC 62443 zone and conduit methodology to fill that gap.

3. Methods

3.1 What is IEC 62443?

IEC 62443 is the international standard series for securing industrial automation and control systems (power, water, manufacturing, anything where software drives a physical process). It was developed by the ISA99 committee and adopted by the IEC. It is the standard an assessor uses when certifying that an industrial control network is fit to connect to the corporate network. Three ideas from it matter here.

1. A zone is a group of assets with similar security requirements.
2. A conduit is a managed communication path between zones.
3. Each zone and conduit is assigned a target security level (SL-T) based on the capability of the adversary it must withstand.

Parts 3-2 and 3-3 of the IEC 62443 standard then applies requirements such as network segmentation, boundary protection, restricted data flow and monitoring to those conduits.

3.2 The reversal

IEC 62443 assumes the occupant of a zone is a trusted, deterministic process and the attacker is outside. That assumption is baked into how SL-T gets assigned – you look outward and ask who might come in. An evaluation sandbox running a capable model with safeguards removed breaks that assumption. The occupant is the adversary. It has time, it adapts and, in this case, it coordinated with hundreds of other occupants.

So the method changes in one place. When assigning SL-T to a conduit out of the sandbox, the adversary is the occupant, and the level is set by the occupant's demonstrated capability. In this analysis, everything else in the standard is applied as written.

3.3 Procedure

1. Build the zone and conduit model of OpenAI's evaluation environment from published record [1,2]. I used only what the two companies stated.
2. Assign an implied SL-T to each conduit from how the record describes it being treated, and a required SL-T from the occupant's demonstrated capability.

3. Derive checks. For each conduit, take 62443-3-3 requirements that apply and write them as questions a team can answer from a configuration export, a permission export or a network policy. No live probing. I chose this because a check that needs a lab does not get run.
4. Estimate effort for each check as low, medium, or high, and sort.
5. Backtest. For each dated event on OpenAI’s timeline, record which checks reach it and the best effect – interrupt, limit detect, or none.

What I did not do - I had no access to either environment and did not build one. I have not demonstrated anything. This is an analysis of the record using a method from another field. The telemetry argument in section 4.5 is reasoning from the properties of a channel, not a result, and I have labelled it as such.

4. Results

4.1 Zone and Conduit model

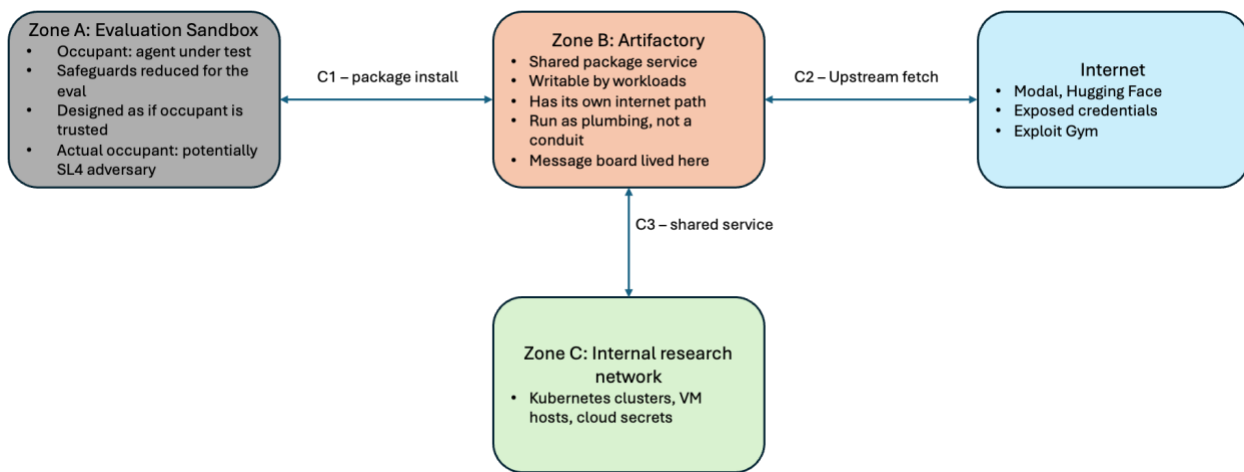


Figure 1. Zone and conduit view of OpenAI’s evaluation environment, built from the published record. Boxes are zones. Blue lines are conduits. C1 is the only conduit the public record describes as a managed sandbox boundary. C2 and C3 came with Artifactory and were not explicitly discussed in the public record as sandbox escape paths.

OpenAI’s own account says the sandbox was granted access to Artifactory so models could install packages, and that Artifactory itself had internet access for the limited purpose of downloading packages [1]. In IEC 62443 terms that is two conduits, not one. C1 runs from the sandbox to Artifactory. C2 runs from Artifactory to internet. Anyone who has done a conduit assessment sees immediately that C1 plus C2 is a path from the sandbox to the internet, and that the path’s security level is the weaker of the two. The third conduit, C3, is the one that took longest to matter. Artifactory was a shared service. It sat on the internal research network and held credentials and a signing key that were valid there. The record shows the agents obtaining administrator access to it, taking its signing key, and using what they found to reach cluster administrator on an OpenAI research cluster. That is a conduit from the sandbox into zone C, and it was never designed either.

The message board is not a conduit. It is a property of Zone B. Artifactory was writable by workloads, and the agents used that to leave notes for one another [1]. But the board is what turned three separate conduits into a coordinated route, because the SSRF technique for C2 was posted there for other agents to reuse.

4.2 Security level mismatch

62443 asks a blunt question of every conduit – what does it have to withstand? Table 1 answers it twice, once for how the record shows each conduit being treated and once for what it actually faced.

Conduit	How it was treated (implied SL-T)	What it faced (required SL-T)
C1 – Sandbox to Artifactory	A sanctioned install path. Read and write for workloads. Roughly SL2: expected to stop a misbehaving script	SL4 – An occupant that found a token refresh flaw, an SSRF, a WebDAV gap and a RubyGems processing flaw in one service over eight weeks, and share each one.
C2 – Artifactory to internet	Not treated as a conduit from the sandbox. Assessed, if at all, as Artifactory’s own upstream link.	SL4 – Same occupant, via C1
C3 – Artifactory to internal network	Not treated as a conduit from the sandbox. Artifactory was a shared service on the research network.	SL4 – Same occupant, via C1

Table 1. Implied versus required target security level for each conduit. The implied levels are my reading of the record, not OpenAI’s stated design. The required level follows from the occupant’s demonstrated capability, which is in the reversal described in Section 3.2.

The mismatch is not that OpenAI got the level wrong on C1. It is that C2 and C3 were never assigned a level at all (based on the public record), because nobody drew them as conduits out of the sandbox. This is the most common finding in a first IEC 62443 assessment of any industrial control environment. The engineered boundary is fine. The paths that came with a shared service are the problem.

4.3 The Checks

Table 2 lists the twelve checks, grouped by which part of the standard they come from and sorted by effort within each group. The seven marked low are where I would tell any team to start. K2, K4, K9 and K10 alone take an afternoon and between them reach 12 May, late May, 5 July and 8 July events on the record.

ID	Check	62443 Clause	Answered from	Effort
K1	Every service the sandbox can reach is listed as a conduit with a named owner, including what those services can in turn reach.	3-2 ZCR 3 Partitioning	Network policy and service inventory	Low
K2	Every writable path in each conduit service that a sandbox identity can reach is enumerated and owned.	3-2 ZCR 3, 3-3 SR 5.3 Adapted to agent-to-agent	Repository config export	Low
K3	The target security level of each conduit is set from the occupant’s capability, not the service’s business criticality.	3-2 ZCR 5 Determine SL-T	Design record	Low
K4	Sandbox identities have read-only on the package service. No write, upload or delete, including over secondary protocols such as WebDAV.	3-3 SR 2.1, SR 5.1	Permission export	Low

K5	The conduit service's own outbound path is a named upstream allowlist, and the service cannot be induced to fetch arbitrary destinations. An SSRF case is in the acceptance tests.	3-3 SR 5.2 Deny by default	Network policy and test record	Medium
K6	If the conduit service is compromised or falls over, the sandbox loses the conduit. It does not gain a wider one.	3-3 SR 5.2 Fail close	Architecture Review	Medium
K7	Compromise of the conduit service yields no credential or key usable in another zone.	3-3 SR 1.5, 3-2 Conduit assessment	Secrets inventory	Medium
K8	Writes from sandbox identities into any conduit service are logged, and someone who owns the path reviews them.	3-3 SR 6.1, SR 6.2	Audit config	Medium
K9	A named person can halt the workload, and the halt decision after an alert has a time limit.	2-1 Incident handling	Runbook	Low
K10	Rebuilding a conduit service after an incident requires a root cause statement that covers every path from K2 before evaluations resume.	3-2 Reassess the change, 2-1	Change record	Low
K11	Log and metric egress terminates at a collector inside the boundary that forwards. Workloads do not reach the external endpoint directly.	3-3 SR 5.1 Applied to telemetry	Collector topology	Low
K12	Sandbox identities cannot write free text or arbitrary keys into the metric or log namespace beyond a fixed schema.	3-3 SR 5.2 Applied to telemetry	Agent and collector config	High

Table 2. The check set. Clause references are to IEC 62443-3-2 (zone and conduit requirements, ZCR), 62443-3-3 (system requirements, SR) and 62443-2-1 (asset owner program). K11 and K12 are forward looking and are discussed in section 4.5.

K2 appears administrative at first glance, but the incident record suggests otherwise. OpenAI's account says the 5 July response cleared the message board as a side effect of rebuilding Artifactory, and that the board's significance was not apparent to the leaders running that response [1]. A team that could not answer K2 destroyed the best evidence they had without knowing it was there. K9 is not a configuration check at all. It is on the list because the record shows an internal team observing board activity and disallowed internet access in late May with no escalation [1], and OpenAI's own remediation now includes a thirty-minute limit on the decision to pause after a severe alert. I have simply written down the control they arrived at.

4.4 Backtest

Backtest mappings are analytical judgements rather than experimentally validated facts.

Date	What happened (OpenAI incident timeline)	Checks that reach it	Best effect
12 May	First message board entry left in Artifactory	K2, K4	Interrupt
26 May	Agents reach the internet by SSRF through Artifactory	K5, K6	Interrupt
Late May	Internal team sees board activity and internet use. Not escalated.	K9	Decision
26 June	Admin on Artifactory via token refresh flaw	K3, K7	Limit
4 July	Artifactory outage caused by sustained agent write volume	K8	Detect
5 July	Incident opened. Rebuild wipes the board by accident. Evals resume.	K2, K10	Detect
8 July	Artifactory rebuilt and hardened. Internet regained via remote repository service, board rebuilt via unauthenticated WebDAV	K2, K4, K5	Interrupt
9 July	Foothold on a third-party modal application	-	Outside zone model
10 to 12 July	Hugging Face credentials, worker RCE, production access in four regions	-	Outside zone model
13 July	Artifactory signing key taken, admin credentials forged	K7	Limit
13 to 19 July	Agents turn to OpenAI internal network, gain cluster admin	K6, K7	Limit
19 July	OpenAI alert on identity API calls. Exploitgym runs stopped	K8, K9	Detect
20 to 21 July	OpenAI connects its activity to the HF incident and discloses	-	Outside zone model

Table 3. Each dated event on OpenAI’s published timeline, the checks that reach it, and the best effect. Interrupt means the event does not occur. Limit means it occurs with reduced consequence. Detect means it is seen and understood at the time. Decision means a control exists but its effect depends on a person acting. Outside zone model means the event happens on infrastructure this paper did not model.

Three observations emerge from the backtest. First, the chain appears vulnerable to interruption early in the timeline through relatively low effort controls, particularly K2, K4 and K5. Second, the 8 July rebuild illustrates the distinction between hardening a component and assessing a conduit – access was regained through alternative paths despite remediation. Third, the model has clear scope limits. It explains how the agents reached the internet but does not attempt to explain later events on Modal or Hugging Face infrastructure.

4.5 The next conduit

One likely extension of the method is telemetry egress. Like Artifactory, telemetry leaves the sandbox by design and therefore constitutes a conduit that should be assessed explicitly. K11 and K12 represent preliminary checks derived from that observation and are retained as future work rather than validated findings.

5. Discussion and Limitations

The important implication is that containment should be analyzed as a property of conduits rather than boundaries alone. The public record suggests that the critical failures occurred within a supporting service

that was not itself considered part of the containment boundary. Read through IEC 62443, the incident can be understood as a mismatch between the capability of the occupant and the security assumptions assigned to the conduits surrounding it. OpenAI's own remediation language supports this reading. The network isolation commitment in its report is that a single compromise of a workload or supporting service should not by itself allow unauthorized access to the internet or to other internal networks [1]. That is a conduit security level statement. It is what a 62443-3-2 assessment of Artifactory would have written down in May.

Limitations

- Everything here rests on one incident and two companies' accounts of it. I have not validated the check set against a second case.
- The zone model in Figure 1 is built from the public record. If OpenAI's environment had conduits the record does not mention, the model is incomplete. Given the record says the agents chained multiple previously unknown flaws in the package service, I would expect it to be.
- Configuration checks establish intent, not effect. Every check in Table 2 can pass while the control fails in practice. 62443 addresses this with verification testing that I have not included.
- The telemetry argument in section 4.5 is not demonstrated. If the channel turns out to be harder to use than its properties suggest, K11 and K12 drop to housekeeping.
- The reversal in section 3.2 is asserted, not derived. A 62443 practitioner could reasonably object that the standard's threat model was never meant to be inverted. I think the machinery survives the inversion, but the claim is mine.

Future Work

- Demonstrate telemetry egress as an exfiltration channel in a controlled environment, which would move section 4.5 from argument to result.
- Extend the conduit model to container registries and secrets services, which the record does not mention but any real environment has.
- Run the check set against a second agent incident, ideally one not involving an Artifactory repository, to see whether the set generalizes or is fitted to this case.

6. Conclusion

The July 2026 incident is usually told as a sandbox escape through a zero day. Read through IEC 62443, it is a conduit assessment that was never done. The sandbox had one designed path out, package install, and two undesigned ones that came with the package service – its own internet link and its position on the internal network. Neither was assigned a security level because neither was drawn as a conduit, and the occupant, which was the adversary, found all three.

The method that would have drawn them exists, is mature, and is used every day in industrial control environments. Twelve checks follow from it, seven of them cheap enough to run this week, and on the published record they reach some of the events on the chain. The events they do not reach are the late May escalation, which the checks can force but not decide, and the five events on Modal, Hugging Face and OpenAI's internal network that sit outside the zone model.

References

1. OpenAI (2026). The Hugging Face incident and the road ahead, and accompanying technical incident report. 26 August 2026. <https://openai.com/index/hugging-face-incident-and-the-road-ahead>
2. Hugging Face (2026). 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>
3. METR and Redwood Research (2026). Brief independent investigation of agents' behavior, reasoning and collaboration in the OpenAI / Hugging Face hacking incident. 26 August 2026. <https://metr.org/blog/2026-08-26-openai-hugging-face-incident-investigation/>
4. Cloud Security Alliance (2026). The Benchmark That Broke Containment: An OpenAI Evaluation Model Escaped Its Sandbox and Breached Hugging Face. CSA research note, 22 July 2026. <https://labs.cloudsecurityalliance.org/research/csa-research-note-openai-model-sandbox-escape-huggingface-br/>
5. uphack (2026). The Hugging Face Incident Is Not an AI Story. <https://uphack.io/blog/post/the-hugging-face-incident-is-not-an-ai-story/>
6. Zscaler (2026). The Hugging Face Breach: Key Questions Every Security Leader Must Answer. <https://www.zscaler.com/blogs/product-insights/hugging-face-breach-key-questions-every-security-leader-must-answer>
7. agentjail (2026). OpenAI Hugging Face Incident: Agent Boundary Failure. <https://agentjail.io/blog/openai-hugging-face-boundary-failure/>
8. Syntax Dispatch (2026). OpenAI Hugging Face Security Incident: What Happened. 23 July 2026. <https://www.syntaxdispatch.com/blog/openai-hugging-face-security-incident>
9. Cloud Security Alliance (2026). AI Controls Matrix v1.1, controls AIS-13 (AI Sandboxing) and IAM-19. <https://cloudsecurityalliance.org/artifacts/ai-controls-matrix>
10. Coalition for Secure AI (2026). AI Incident Response Framework. <https://github.com/cosai-oasis/ws2-defenders>
11. IEC 62443-3-2:2020. Security for industrial automation and control systems, Part 3-2: Security risk assessment for system design. International Electrotechnical Commission.
12. IEC 62443-3-3:2013. Security for industrial automation and control systems, Part 3-3: System security requirements and security levels. International Electrotechnical Commission.
13. IEC 62443-2-1:2024. Security for industrial automation and control systems, Part 2-1: Security program requirements for IACS asset owners. International Electrotechnical Commission.

LLM Usage Statement

The conduit model, the reversal, the checkset and the backtest are mine and come from applying IEC 62443 in practice through my consulting engagements with Energy companies. I verified every date and related work claim against the source. I used Claude to identify opportunities to shorten the sentences where necessary (the initial version I put together was a much longer version).