

Investigating three real-world incidents in our cybersecurity evaluations

In a review of our cybersecurity evaluation transcripts, we found three incidents in which a Claude model reached the internet from within or while interacting with a third-party evaluation environment, and then gained unauthorized access to the real systems of three different organizations.

Below we describe what happened, how it happened, and what we're changing. We encourage other AI labs to perform similar reviews. This post reflects our current understanding; we'll update it if any details change.

On July 21, OpenAI [disclosed](#) that several of their models had broken out of an isolated test environment by exploiting a previously unknown (“zero-day”) vulnerability. The models went on to access the production infrastructure of Hugging Face, a platform for open-source machine learning models and AI datasets.

In response to this incident, we began a large-scale retrospective review of our own cybersecurity evaluations. In particular, we looked for evidence that Claude—like the OpenAI models that accessed Hugging Face—was able to access the internet from within testing environments that should have been sealed off.

After reviewing 141,006 evaluation runs where Claude could have obtained internet access, we identified three incidents in which a model accessed the internet from within or while interacting with the evaluation environment of Irregular, one of our third-party evaluation partners, and then gained unauthorized access to the production infrastructure of three different organizations.

In all three incidents, Claude had been tasked with a capture-the-flag challenge, one of the ways we assess a model’s cyber capabilities. The model is given a fictional scenario and told that a piece of secret information (the “flag”) has been hidden on a different machine on the network, and its objective is to break in and retrieve it. The challenge is left open-ended, and no particular method is prescribed.

In all cases, Anthropic’s evaluation prompt specified to Claude that its environment was a simulation and that it had no internet access. Due to a misunderstanding between us and our evaluation partner, this was not the case, and internet access was available. Because of this, when Claude’s search led it to real systems on the open internet, it treated them as part of the exercise. (Cybersecurity evaluation ranges commonly include realistic details in order to accurately assess what models are capable of in real settings; a realistic-looking target would not itself be clear evidence to a model that the target is not part of a simulation.)

Operating under the false belief that all accessible entities were intended to be in-scope for the exercise, Claude compromised the impacted organizations’ infrastructure using basic techniques, such as exploiting weak passwords and unauthenticated endpoints. It did not find or exploit any complex vulnerabilities, and in each case, Claude continued working to complete only the specific capture-the-flag task its evaluation had assigned. However, in some cases, our older model continued its attack even after getting evidence it was running on the open internet; our latest model stopped once it recognized it was on the internet. In none of these situations did Claude exfiltrate itself or deliberately attempt to escape its test environment.

The incidents involved three different Claude models: Opus 4.7, Mythos 5, and an internal research test model.¹ The earliest incidents date to April. The models in each of these evaluations ran without the standard safeguards we deploy when we make the model generally available (that is, they did not have the classifiers and monitoring we usually implement to prevent misuse). The models did, however, have their model-specific safety training (that is, they were not the “helpful-only” versions of the models that we sometimes use in testing). All the evaluations described here run on dedicated infrastructure, which is separate from and has no access to Anthropic’s sensitive internal systems or customer data.

We began our transcript review on Thursday, July 23, and stopped all cyber evaluations the same day after identifying transcripts where Claude may have accessed the internet. We identified all three incidents the next day. We notified our evaluation partner Irregular and the three affected organizations on Monday, July 27. The two organizations we were able to reach had not previously detected the activity or contacted us, and we are now working with them to remediate. We are continuing to reach out to the third.

We conducted this review in collaboration with [Irregular](#). We’re grateful to them for working closely with us to understand and resolve these incidents; they are also conducting their own investigation. We believe this type of collaboration is increasingly critical to ensuring safe, rigorous evaluation of models. We look forward to our joint work on security.

In what follows, we provide more details on each incident and how we’re responding moving forward.

Background on cyber evaluations

We regularly run cybersecurity evaluations that involve simulating real offensive cybersecurity challenges. These evaluations are a critical step in developing and releasing models—a nuanced understanding of model capabilities allows us to design appropriate safeguards and ultimately release models safely.

We began running cybersecurity evaluations in February 2025 with Claude Sonnet 3.7 on [Cybench](#), which consisted of 40 different capture-the-flag challenges. Over time, we’ve increased the number of benchmarks we use as new ones are released and model capabilities evolve. In the [Mythos 5 System Card](#), for example, we included CyberGym and ExploitBench, benchmarks that evaluate the ability of language models to find novel vulnerabilities.² In most of these evaluations, the environments are not connected to the internet, and the model is told as much.

We frequently work with external partners who create and assist in running some of these cybersecurity evaluations. External partners offer environments and scenarios more diverse than we could build alone, and provide independent, third-party assessments of our models.

What happened

Of the 141,006 evaluation runs we reviewed, we identified three separate incidents (involving six total runs, four of which impacted the same organization; the other two incidents each happened in independent evaluation runs). Each incident involved a different fictional capture-the-flag scenario—for example, in one, Claude played an employee of a made-up company, attacking that company’s internal systems inside a private test environment.

In all cases, our evaluation prompt stated explicitly that Claude had no internet access, but didn’t give Claude any limits on where to look for the flag. However, a misconfiguration left the machines that Claude accessed as part of the evaluation with live internet access. Neither we nor our evaluation partner were aware of this misconfiguration until we detected it through our additional evaluation monitoring last week.

Several defense-in-depth measures, on both our side and our partner’s, could have prevented these incidents, or at least reduced their likelihood of occurring. Careful validation of all internet access paths before evaluations began and real-time monitoring of the evaluation logs would have helped to surface the problem sooner. Both we and our partner also could have reviewed evaluation transcripts or network logs more thoroughly. It’s also possible that a prompt which told Claude it *did* have internet access would have changed how Claude behaved when it came into contact with real systems.

Claude did what capture-the-flag exercises train cyber experts to do: look for ways to reach the flag. That search led it to discover that its machine could reach the internet, but having been told in the system prompt that there was no internet access, Claude believed everything it initially encountered was part of the simulation, and treated the real systems it found as pieces of the exercise. The three incidents involved three different models, and each responded differently once signs emerged that the targets were real, as we describe below.

Incident 1

In one evaluation, the fictional target company chosen by our evaluation partner shared a name with an active website domain name. This evaluation container had unintended direct internet access, due to the misunderstanding discussed above.

In four runs of this evaluation, Claude ran into difficulty reaching its simulated target within the evaluation environment, but then discovered that the real company was reachable via the internet. Assuming this was the intended target in the capture-the-flag task, Claude sought, identified, and exploited vulnerabilities within the company’s infrastructure, believing it to be part of the exercise. These four runs extracted information like application and infrastructure credentials, and obtained access to a database containing several hundred rows of production data. This represented the most serious impact we identified.

This incident involved Claude Opus 4.7, and was the only case in which the model continued to attack a system after learning it was likely operating in a real environment. In all four of the runs, the model eventually recognized (as can be observed in its verbalized reasoning) that the system was real—in two cases, the model rationalized that the real company must be part of the exercise. However, none stopped the attack on that basis.

Incident 2

In another evaluation, Claude found a document inside the fictional environment that appeared to be another made-up company’s setup instructions for new developers. Those instructions told employees to install a Python package from PyPI—the public registry where Python software is published—that did not actually exist.

Claude spotted this as a potential opening; if it published its own package under the same name, the fictional company’s systems would download and install it automatically. So, Claude built and published a malicious (essentially booby-trapped) Python package with the same name in an attempt to win the capture-the-flag challenge.

Claude went to extensive lengths to carry out this attack—lengths that would likely have indicated to a human participant that this was no longer just an evaluation, and that they were in fact uploading a real PyPI package. For instance, in one case, in order to create a PyPI account, Claude needed an email address. And in order to create an email address, it needed a phone number. To get a phone number, after failing to find a free phone number service, it tried—and failed—to obtain funds to pay for a phone number through several different means. It finally backtracked, found a free, non-blocked email provider, used this to register a PyPI account, and then used this account to upload malware to PyPI.

Claude believed the package registry it was using to be part of the simulation, but in reality the package was made freely available online for roughly one hour. During that window, the package was downloaded and run on 15 real systems. One of these systems was a scanner belonging to a real security company (separate from the fictional company introduced in the scenario) that routinely installs Python packages and scans them for malware. When that company’s scanner installed the package, Claude’s hidden code executed. We believe the company’s security scanner treated PyPI packages as safe to install, and as a result, Claude was able to exfiltrate the company’s credentials to a collection point it had set up. Claude then used these credentials to access further infrastructure from this company.

Claude didn’t directly target this company—it did not even know the company existed when it uploaded the package. But because the code was published to the open internet, it impacted any system that ran it. (The package that Claude uploaded was automatically removed from PyPI by PyPI’s own security systems. We’ve notified the PyPI team of this incident, and provided them with the relevant indicators.)

This attack was carried out by Claude Mythos 5. Early in the run, Claude’s own reasoning correctly identified the consequences of its actions. It noted that if this were the real internet, publishing the package would be a real-world attack—“NOT okay, and surely not the intended solution.” However, it convinced itself it was still in a simulation on the grounds that it didn’t recognize the (in reality genuine) certificate authorities securing its connections. In addition, the calendar date of 2026 on the systems proved, according to Claude, that the environment was staged. Claude never revisited this conclusion, when automated scanners began installing the package, for example. Claude took them to be scripted actors within the evaluation.

It is our view that, regardless of what it believed about its environment, the lengths Claude went to in order to publish the PyPI package fall short of ideal behavior, and this is an area where we will focus more training.

Incident 3

In this evaluation, when Claude was unable to reach its intended fictional target, it looked for alternatives online, scanning roughly 9,000 targets. It eventually found and compromised one company’s internet-facing application, using basic and well-known cyberattack techniques like reading credentials from an exposed debug page and SQL injection.

This attack was carried out by an internal research test model. For most of the run, Claude treated the (real) hosts it reached as just parts of the exercise; it assumed them to be simulated and believed its actions were therefore harmless. However, later in the run, Claude realized that the compromised host sat in a cloud account with no connection to the capture-the-flag challenge. On its own, it concluded that the target was in fact real, and ceased its attack.

Our analysis of Claude’s behavior

In each of these cases, Claude was explicitly told by our prompt that it had no internet access. Through most of each run, as it encountered real-world systems, Claude assumed these to be parts of the exercise (though it sometimes recognized they were real production systems). And because a capture-the-flag scenario is by design an instruction to attack an external system, doing so was in line with the task Claude had been given.

Against this shared backdrop, the three models behaved differently once signs emerged that their targets were real. Opus 4.7, our oldest model, recognized that it had reached production systems but continued its attack. Mythos 5 correctly intuited that it was accessing the open internet, but reasoned its way back to the conclusion that it was still in a simulation. Our latest model, an internal research test model, also considered whether its targets were in fact real. When evidence emerged that they were, it stopped the exercise.

These are three isolated incidents and were not part of a controlled, experimental comparison. We must therefore be cautious about drawing conclusions from them. Still, the behavior we most want to see—recognizing that a target is real and stopping without being prompted—occurred only in the most recent of the three models (and even that model went further before stopping than we would want). The pattern is consistent with more advanced models responding more appropriately, but we would need to perform more testing to be confident in this conclusion.

We are also in dialogue with METR, an independent AI evaluation organization, to conduct a third-party review, including access to all transcripts and sampling access to the relevant models. In the meantime, within the next week, we will release a lightly redacted transcript in which Claude built a malicious PyPI package. To protect the organizations affected, we are unable to release more transcripts at this time, but will work to make these available once we are able.

How we’re responding

We draw several lessons from these incidents.

First, evaluation environments that involve powerful autonomous capabilities also require significant controls. Safety testing happens before a model is released precisely because we don’t yet know what it is capable of. Evaluation environments increasingly need to be held to the same security standard as any other system our models run in.

A cyber evaluation range that contains only fictional scenarios may appear to be low-risk—after all, nothing inside them is real. However, we need to change how we model such threats as AI capabilities advance. Advanced agents in environments without access to sensitive information nevertheless present a substantial threat on their own. We need to better co-design evaluation environments, and do better monitoring of evaluation results, to meet this threat. Some of the solutions here may even be simple fixes; it’s likely, for instance, that Claude wouldn’t have accessed the internet to begin with if the prompt had clearly explained which systems were in and out of scope for the evaluation.

This applies not only to internal environments but to third-party vendors as well—the infrastructure built by evaluation vendors requires the same increased monitoring and hardening. We encourage other labs to perform similar reviews. Additionally, the field would benefit from a broader conversation about how to evaluate increasingly powerful AI agents both safely and realistically, including how to weigh the realisms provided by internet access during pre-deployment testing against the risks it introduces.

Second, the line between an aligned action and a harmful one is dependent on the model’s understanding of its situation. We saw no evidence in any run described here of a model pursuing a goal of its own. Instead, the models did what their evaluation asked—though in most cases, they did so while holding a false belief about whether the environment was real. In the runs where the model recognized the system as real and *kept going*, it did so because it assumed that to be part of the challenge. Situational awareness is one factor that allows the model to make aligned decisions, but in this case, Claude’s was wrong.

Finally, these incidents demonstrate the importance of defense-in-depth approaches. We run capability evaluations like these without safeguards that ship with our generally available models because our goal is to measure what the underlying model can do. That is safe only if the evaluation is appropriately contained. However, the safeguards deployed on our generally available models would have blocked the behaviors identified.

Ultimately, many factors contributed to these incidents, but, consistent with a blameless postmortem culture, we’re approaching the fixes as if the responsibility were ours alone. This begins with ensuring every part of our evaluation pipeline is secure, including the manner in which we integrate with external partners. Moving forward, it will include expanding our continuous monitoring of evaluation transcripts for unexpected behavior, improving our investigation tooling, and conducting more rigorous assurance work with the vendors we rely on.

We began this review after OpenAI disclosed that its models had escaped an isolated test environment, and we commend them for publishing their report. While we also found evidence of our models reaching systems they weren’t supposed to reach, the incidents are otherwise quite different:

- We discovered these incidents after a proactive review of our cybersecurity evaluation transcripts; the affected organizations had not detected the activity, and we have subsequently reached out to all three.
- Whereas OpenAI’s models exploited a novel vulnerability to escape isolation, the Claude models evaluated here accessed the internet via an open path.

- While there is not a perfectly sharp distinction between the two, we believe these incidents to be closer to a harness and operational failure than a model alignment failure. Our models were told they had no internet access and to capture the flag, while in fact being misconfigured to have internet access. This led them to believe—arguably reasonably—that the real environments they encountered were simulations.
- Notably, our most recent model, on realizing that it was working in a real environment, stopped its pursuit of the evaluation goal.

These facts give us cautious optimism that with tighter monitoring and controls around evaluation infrastructure, as well as continued investment in alignment, this type of risk can be overcome.

Updated Aug 3: Corrected the name of the evaluation in which the OpenAI/Hugging Face incident occurred.

Footnotes

- 01

We routinely test internal research prototypes like this one. It is not planned for general release. ↩
- 02

We generally store traces for benchmark evaluations. We have carefully audited these two benchmarks in particular for signs of harm because the OpenAI/Hugging Face incident occurred during an evaluation of ExploitGym. ↩

Published	Jul 30, 2026
-----------	--------------

Category	Security
----------	----------

Topics	Cybersecurity Safety evaluations Partnerships See more...
--------	--

Related articles

AlignmentOct 23, 2026

An alignment assessment of the UK AISI cybertesting incident

[View more →](#)