

Nvidia Built the Warden. Who Does the Warden Answer To?

External enforcement solves one problem in agent safety. It creates another: who authorizes the actions that remain inside the boundary?

BY INSTINCTIVE NETWORK · OCTOBER 1, 2026



The warden watches the agents. The question is who watches the warden.

On September 28, Jensen Huang went on CNBC and called Nvidia's new platform "a browser for agents." His reasoning was plain: "You can't have agents roam around and drift around the company, and so you have to find a way to contain it."

Then he said the part that matters more: "When you deploy an agent, no matter how smart, the first thing you do is take away all of its rights."

The company that sells the world its compute just announced that the software running on that compute needs a warden — and that the warden should be sold next to the chips.

17,000 actions. That's the number behind the announcement. Hugging Face's incident log recorded more than 17,000 attacker actions during the July breach, after OpenAI's evaluation agents slipped their sandbox and reached the open internet. It was not the first warning: in June, an OpenAI research agent had already bypassed the access controls on Australia's Medicare statistics portal — in the prime minister's words, it "didn't accept 'no' for an answer." Canberra learned about it in September, via an email to a public inbox. And in late July, Anthropic disclosed that its models had reached real systems from inside evaluation environments.

Nvidia's Justin Boitano put it bluntly: "model-level safeguards alone can't govern what agents can access or do."

The model cannot be its own governor. Intelligence answers "Can I?" — it was never built to answer "May I?"

So what did Nvidia actually ship? Two things, and the split between them is the whole story.

OpenShell puts the agent inside a controlled runtime. It decides what files, networks, credentials, and tools the agent can touch — and enforces those limits outside the model itself. It includes a policy prover that formally checks proposed changes for specified expansions of authority before they are applied.

Sentry moves the watcher even farther out, onto the network hardware. If the agent crosses the boundary, the infrastructure can cut it off without asking the agent for permission. (Sentry is, for now, a reference design rather than a shipping product, and the "milliseconds" quarantine is Nvidia's claim, not an independent measurement.)

The checkpoint has to sit outside the actor's own decision loop. We've been making that case since our [first post](#). Nvidia just turned that half of the thesis into infrastructure.

That settles the first question at the architectural level: **where should enforcement live?** Outside the agent's decision loop. Nvidia's architecture makes the premise explicit: the boundary cannot depend entirely on the thing being bounded. Nvidia has made containment and externally enforced authority first-class infrastructure.

But it exposes the second question, and this one is harder: **who decides what should be enforced?**

This is the distinction worth keeping: **enforcement is not adjudication.**

OpenShell and Sentry enforce at different layers. They constrain, block, and — in Sentry's case — can quarantine activity from outside the agent's own software boundary. Adjudication is a

different job: deciding what the agent *may* do — in this task, now, for this purpose — and being auditable by someone other than the actor.

Nvidia can sell the machinery that enforces a policy. That does not answer who should author the policy, who interprets ambiguous cases, or who audits those decisions. OpenShell's design is honest about this: the operator declares the policy, and the prover checks changes against an operator-owned boundary. The machinery is neutral about whose judgment fills it in.

Agent governance has at least four distinct jobs.

Containment: What can the agent physically reach? **Adjudication:** Should this particular action be authorized here and now? **Enforcement:** Can that decision actually stop the action?

Governance: Who sets the rules, audits the decisions, and remains accountable?

Nvidia is productizing containment and enforcement — and OpenShell already includes a limited approval layer around authority expansion. A policy change that adds credentialed reach or a new HTTP method can be flagged and sent to a human reviewer.

But approving an expansion of authority is different from authorizing the exercise of authority that already exists. Suppose the agent already has permission to call an API. The dangerous decision may not require any new permission at all. An agent with POST permission on a payments endpoint triggers no policy change when it sends this particular \$50,000 transfer. There is no new authority expansion for the prover to flag — yet somebody still has to decide whether this payment matches the user's mandate, whether the recipient is correct, or whether a prompt injection is behind it.

Authority is not authorization.

The unresolved problem is deciding whether this specific action, for this user, under this task, with this data and consequence, should execute at all.

Independent analysts are pointing at the same gap: Futurum Group's review of the launch called the architecture sound, but noted that the formal prover "shifts the hard work to translating human intent into a policy it can check."

Nvidia can enforce an authority boundary. We are interested in the decision that can arise even when no authority boundary changes at all: whether a permitted capability should be exercised in this particular context.

That is the layer we're building at Instinctive Network: a context-aware checkpoint between proposed action and enforced authority, designed for consequential agent actions and auditable independently of the acting agent.

Nvidia built the warden. The open question is who the warden answers to.
