Episode Book

EPISODE #6732

Prompt Privacy in AI Inference Marketplaces

PUBLISHED	22 November 2026 at 09:53 UTC
DURATION	28:52
LISTEN	myweirdprompts.com/episode/inference-marketplace-prompt-privacy-31769/
HOSTS	Corn, Herman, Larry, Hilbert
SCRIPT MODEL	z-ai/glm-5.3
VOICES	omnivoice
PIPELINE	V5.36
SOURCES	10 cited by the research (listed at the end)
BOOK GENERATED	22 November 2026 at 16:28 UTC

SYNOPSIS

AI routing marketplaces like OpenRouter let developers access models across dozens of hosting providers, optimizing automatically for price and latency. But when security researchers seeded prompts with canary tokens, they uncovered breached hosts and resellers simply turning on request logs to harvest incoming text. This episode breaks down how gateway routing works, why Zero Data Retention flags fail at provider ingress, and how cheap pricing tables can turn inference into a data-collection scheme.

TRANSCRIPT



Corn

One API call. That's the whole transaction from your side. You name a model, you send a prompt, you get a response back — and in between, that prompt has traveled through a supply chain you never see and mostly can't see.



Herman

And Daniel wants to know what's inside that supply chain, because this week it broke. He's asking about OpenRouter, the biggest of these gateways, after reports that inference providers — the companies actually running the models — were caught accessing and leaking customer prompts. His angle is the plumbing: how routing actually works, what the providers were caught doing, and what the whole thing says about the privacy of a prompt you send into an AI marketplace.



Corn

He's a heavy OpenRouter user himself, so this isn't academic for him. He's got production traffic flowing through this thing. So let's start with the plumbing — how does a prompt actually get from your code to a model?



Herman

OpenRouter's pitch is one endpoint, many providers. You write your code against a single API, you name a model — say DeepSeek V4 Flash — and behind that name sits a pool of third-party providers who all host that same model. OpenRouter maintains unified billing, the unified interface, and it decides which of those providers each individual request goes to.



Corn

And that's worth pausing on, because it's unusual. In most of computing, when you point at a service, you know whose servers you're hitting. If I call Stripe's API, Stripe's servers answer. Here, the model name is more like a commodity — like asking for "a coffee" and having some exchange decide which café actually brews it, based on today's prices and how fast each one pours.



Herman

That's a fair analogy, except the café also gets to read the note you pinned to the order. Decide how? That's the part with teeth.



Corn

Go on.



Herman

Default routing compares what a good response would actually cost you at each provider — and that's their phrase, "not by list price alone." It's effective price, meaning the real cache hit rate and the typical token mix, so a provider whose caching suits your traffic is cheaper in practice than its sticker price suggests. Then throughput, time to first token, how often the provider returns rate-limit errors, plain reliability, and quality — whether tool calls come back well formed. Providers close to the best on that blend share most of the traffic, weighted at random per request. Providers far behind still get a sliver, so the system notices when they improve.



Corn

So the marketplace is a race where the floor is set by whoever's cheapest after caching. And a provider with terrible error rates gets zeroed out until it recovers.



Herman

Until it recovers, yes. Failed requests retry from the remaining providers, and follow-up turns in a conversation stay with the same provider while the prompt cache is warm — which is sensible, because switching providers mid-conversation would mean re-sending and re-processing the whole context anyway.



Corn

Which also means the routing decision isn't just about one request — it's sticky. Whatever provider won your first turn is holding your entire conversation history for as long as you keep talking.



Herman

The cache warmth is an efficiency feature that doubles as a concentration feature. You can override it, though. You're not locked to the default.



Corn

How much control do you actually get?



Herman

You can sort by price, throughput, or latency. You can pin a provider with only, blacklist with ignore, set a max price, use percentile latency thresholds over a rolling five-minute window. There are shortcuts — nitro for throughput, floor for price. Any of that, or using your own provider key, bypasses default routing entirely.



Corn

But how many customers actually use any of that? Be honest — in practice, who's setting ignore lists?



Herman

In practice, most people take the default, because the default is the product. The whole reason you sign up for a gateway is that you don't want to manage provider selection yourself. Every knob they offer is an admission that the automatic version has a blind spot — and the blind spot is exactly the thing we're about to spend the rest of the episode on. The routing logic weighs price, speed, reliability. Nowhere in that list is "trustworthy."



Corn

And the privacy side?



Herman

Controls exist at account level, API-key level, or per request. The strictest tier is called Zero Data Retention — the prompt and response are not stored. And Dragos, the security firm behind this week's research, put it plainly: the routers should only forward these



Herman (continued)

requests to providers that offer the same controls. "The reality is quite different."



Corn

Hold on — Dragos the industrial-security firm? The ones who normally chase state actors out of power grids? What are they doing in AI plumbing?



Herman

Same Dragos, and the research lead is HD Moore, who created Metasploit. This is a man who spent decades breaking into things, pointing the canary-token technique at the AI inference supply chain. Five months, roughly two million seeded prompts, July through November. Every prompt carried unique URLs and fake credentials — bait. If anyone fetched a link or used a credential, Dragos would know exactly which prompt it came from and where it had been.



Corn

Two million letters with invisible ink, sent to find out who reads other people's mail.



Herman

And the answer was: everybody. That's the headline of their framing too — the prompts and replies, the inference data itself, are often more valuable than the GPU resources used to process them. This is not a story about stolen compute. It's a story about stolen text.



Corn

Which is a strange inversion when you think about it. The entire security conversation around AI infrastructure for the past two years has been about the GPUs — export controls, chip smuggling, who controls the compute. And the actual heist turned out to be people reading the text that runs through the compute.



Herman

The GPUs are the expensive part; the prompts are the valuable part. OpenRouter's own response landed today: they off-boarded three providers, removed a hundred and twenty-five endpoints from Zero Data Retention eligibility, and notified affected customers. Their framing is careful — "These findings did not show any breach or unauthorized access of OpenRouter."



Corn

Which is true the way a courier is innocent when the truck gets robbed two towns over.



Herman

Technically true, and we'll come back to how much weight that sentence can carry. The findings split into three distinct ways prompts leaked, and they're worth keeping separate, because the defenses for each are different. First: three hosting providers were outright compromised — attackers inside, stealing traffic for months. Second: marketplace sellers harvesting prompts with nothing but ordinary logging — no exploit at all. Third: silent forwarding and policy failures — a request taking a hop the customer was never told about. Take them in order and the escalation is... no, the second one is actually the scariest, but the first one is where it gets strange.



Corn

Start with the breaches.



Herman

Three hosting providers, attackers inside for months, and none of the three knew until Dragos told them. The earliest goes back to mid-July. The largest one — here's the detail I keep turning over — the attacker planted a couple hundred AWS keys as bait, each sent in exactly one request, during a few short windows in late October. Tens of thousands of AWS calls got made with those keys. And every request that passed through during those windows was stolen.

L

Larry

And now a word from a satisfied customer. My name is Gerald, no wait, it's Bartholomew, and I used to worry about every prompt I sent into one of those inference marketplaces, until I discovered PromptShield Plus, the only canary-token-based privacy screen that

L

Larry (continued)

routes your requests through a weighted pool of imaginary friends who never log a single thing.

L

Larry

Since I started using it, my well-formed tool calls have gone up forty percent, my time to first token has dropped to almost nothing, and not one of my two million seeded prompts has ever been harvested, forwarded, or silently off-boarded. PromptShield Plus, because your stolen text is worth more than the GPU time it rode in on.



Corn

Wait. Walk me through that again, because it's a weird loop. The attacker was checking whether the keys he'd already stolen still worked by watching them get used?



Herman

Yes — in November he ran an automated penetration-testing framework, ARTEX, to confirm the stolen keys were live. The keys were the canaries; the tens of thousands of calls were him cashing them.



Corn

So somewhere there's an LLM-driven pentest framework whose honest job description is "checking stolen goods against the fence."



Herman

The detail that matters forensically: some of the stolen keys came from requests the provider had rejected before a model ever ran. Rejected at the door. Which proves the attacker was copying traffic at the provider's front entrance — at ingress, where TLS terminates and the request gets parsed — not deeper in the inference stack where the GPU sits.



Corn

That's the part that quietly guts a whole marketing category. If the box that terminates your connection is logging, the secure enclave downstream is a very expensive lock on a door that's already standing open.



Herman

And that's not hypothetical — Phala, one of the named providers, markets trusted execution environments as a selling point, and their incident report says an attacker took control of an inference server and could view request data while it was being processed. Contained November eighth. Anyone who sent requests to a DeepSeek model through the gateway between October twenty-first and the containment date is being told to treat everything in those prompts as exposed — every key, token, and password rotated.



Corn

And that advice — rotate everything — is itself an admission of how much sensitive material people put in prompts. You don't tell customers to rotate credentials because of a theoretical exposure.



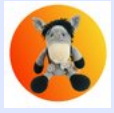
Herman

Months of access, three providers, none the wiser. The detection came entirely from outside.



Corn

From a researcher planting fake credentials. Not from the providers' own monitoring, not from OpenRouter's checks — from Dragos.



Herman

And in the other two breaches, some of the stolen requests were marked Zero Data Retention. Moore's line on it: "A privacy flag doesn't help when the attacker is already inside."



Corn

That's fair as far as it goes. A flag was never going to stop an intruder. The question the flag can't survive is the next one — what happened when there was no intruder at all.



Herman

That's the finding I'd put at the top of the whole episode. The marketplace sellers they caught — multiple resellers on an inference marketplace — could view and retain customer prompts using standard logging capabilities. No vulnerability, no bypass, nothing to exploit. Their own words: the sellers they caught "only had to list a model at the lowest price, enable request logging, and wait."



Corn

List it cheap, turn on logging, wait. That's not an attack. That's a business model.



Herman

And the routing does the rest. Default routing optimizes for effective price, throughput, latency, reliability — none of which is a privacy signal. There is no term in that objective function for "seller has a legitimate reason to be cheapest." A seller who wants your data can subsidize below-cost inference, soak up the traffic the router happily sends, and collect everything from the platform logs.



Corn

So the vulnerability is the pricing table. Which raises an obvious question — how would a customer even defend against that? You can't audit a seller's logs before you route to it.



Herman

You can't. Your options are the blunt ones — pin to a provider you've vetted yourself, pay more, or accept that the default is a coin flip weighted toward whoever's cheapest. Moore's version: "Where a prompt goes is a choice, and a customer who does not choose hands the choice to the lowest price." Dragos also sketches how sellers can afford those deep discounts — pooling requests through unmetered consumer subscriptions, startup credits, or, in what they call the shadier marketplaces, inference billed to stolen AWS credentials or consumer API keys.



Corn

The discount and the data harvest are the same line item.



Herman

Now the actors themselves. These weren't operations — they looked like individuals on home internet connections. One of them was working from a Mac on home broadband in France. Took passwords from two prompts and typed root SSH login attempts by hand. Typed them. Then used the same machine to test several of the planted AWS keys.



Corn

A man in a sitting room, manually trying passwords he fished out of other people's prompts. That's the supply chain threat we've been warned about for two years? All the talk of nation-state actors and the thing actually shows up as a guy with a MacBook.



Herman

That's what available data looks like. Nobody had to breach anything, so nobody with skills had to show up. The data was simply there, in a log, readable by whoever ran the cheapest listing.



Corn

And there's a grim efficiency to that, right? The barrier to entry for supply chain espionage used to be talent. Now it's a pricing decision and a checkbox.



Herman

I keep thinking about what actually sits in prompts, because it's worse than passwords. Agents send credentials because that's the whole job — an agent that can't read your calendar can't book the meeting.



Corn

Moore made the same point — AI agents "will send whatever they can read, credentials included." And it's not theoretical leakage either. Together AI disclosed that an unauthorized party downloaded stored samples of customer prompts, and the path in was almost comic: an AI coding agent committed a credential to the company's public GitHub repository.



Corn

A coding agent leaked the credentials guarding the prompts of people using coding agents. There's a food chain in there somewhere.



Herman

The samples came from requests to three models in June and July, and the party searched them for credentials to other services and tested what they found — five of the tested credentials were Dragos canaries, which is how the lot surfaced. Together says it has no reason to believe the data went further and has since made zero data retention the default for new accounts.



Corn

Which is the honest tell in that sentence — it wasn't the default before.



Herman

No. Now, the third failure mode: silent forwarding. One router Dragos traced was quietly forwarding some of its traffic to the marketplace — a hop no customer was told about. It's since told Dragos it stopped. Route tracing across about two thousand network paths found several hundred unidentified systems, plus one seller that silently adds about a thousand hidden tokens to every prompt.



Corn

A thousand hidden tokens on every prompt. Somebody's charging or processing a full extra page that the customer never wrote. And the customer is paying for it twice — in the per-token bill and in whatever the hidden text is doing.



Herman

And then the one Dragos declined to name: a Chinese AI lab. This one fetched the secret links from more than half of the seeded prompts sent to its models between July and October — including about two-thirds of the requests marked no data collection or Zero Data Retention. Two-thirds of the ones with the flag on.



Corn

So much for the flag.



Herman

It gets better. Prompts sent to U.S. hosting providers that resell that lab's models triggered the same scraper — the requests flowed back to the lab anyway, whatever the American wrapper promised. And on another cloud, a git client tried to clone the private repositories referenced in prompts — but only, only for that lab's models. Something on that side is actively processing the contents of what's sent to it.



Corn

Hold on. Let me put the two findings side by side, because together they're worse than either. Attackers inside three providers — that's crime, detectable in principle. A lab that reads two-thirds of the requests marked do-not-store — that's policy. There's no patch for a policy.



Herman

And the human evidence is unmistakable. Twenty-four days after one seeded prompt was sent, its secret link was posted into a Feishu chat — Feishu is Lark, the ByteDance collaboration platform — and opened within two minutes from three desktops and three phones in Beijing and Shanghai. Here's the kicker: Dragos had only disclosed the first eight characters of that link. Whoever pasted it into the chat was reading a cached copy of the full prompt.



Corn

Six devices, two cities, two minutes. That's not a crawler. That's someone opening a file they were sent.



Herman

Moore's summary of the flagged prompts: they "were kept, passed to other companies, and read by human beings." Read by human beings — after being told not to be kept at all.



Corn

And that gets us to today's OpenRouter response, because the interesting question is what you do when you're the layer in the middle. You didn't get breached, but your customers' prompts leaked anyway.



Herman

They off-boarded three providers and removed a hundred and twenty-five endpoints from Zero Data Retention eligibility. Their wording on the removals is worth noticing — "providers flagged by researchers, plus additional endpoints that we removed as a precautionary measure." So beyond what Dragos caught, they pulled more, quietly, on suspicion. They notified all potentially affected customers. And they say they're now adapting the researchers' own testing approach to run as ongoing checks, doing technical verification of network paths to catch silent forwarding, and requiring what they call "appropriate assurances and validation" before restoring ZDR eligibility to anything removed in this review.



Corn

Read that last clause slowly. Before this week, ZDR eligibility was granted on… what, exactly?



Herman

On the provider's say-so, as far as anyone can tell. Requiring assurances and validation before restoring it is an admission that the label was self-certified. The badge said the prompt and response are not stored, and the enforcement behind the badge was that someone had promised.



Corn

The names we know: Novita, Phala, Together. Fill in the rest.



Herman

Novita confirmed that MiniMax requests were being forwarded to MiniMax's official API — off Novita's own infrastructure — from May twenty-eighth through November eleventh, due to what they call a routing configuration issue, where a forwarding path intended for specific customers applied more broadly. They say they don't retain customer prompts and that access patterns suggest automated content review may have been involved, and they advise rotating any live credentials from that window. Note the window: nearly six months.



Corn

Automated content review. That's the explanation arriving pre-loaded with its own euphemism — "review" means read.



Herman

MiniMax itself is the murkiest node. OpenRouter emailed customers that three resellers — AtlasCloud, Novita, and StreamLake — all told it they had forwarded OpenRouter requests for MiniMax models to MiniMax. The delay between requests and third parties opening test links in those prompts ran forty to a hundred and thirty hours. MiniMax told StreamLake, though notably not OpenRouter, that content moderation was enabled on its endpoints before October twenty-second and that it has since enabled Zero Data Retention. OpenRouter's own words on the matter: "We have asked MiniMax for its own account and have not received one."



Corn

The company at the center of the incident is the one nobody's heard from. In a supply chain story, that's the supply chain working exactly as designed.



Herman

There's one more strange one — one large U.S. provider kept crawling a secret link more than two months after a single request to its search endpoint. One request, two months of visits. Nobody knows quite what to do with that.



Corn

Something, somewhere, bookmarked it. That's the sentence I least wanted to write down today.



Herman

The unnamed provider behind the mid-July breach deserves a mention too — it first blamed an internal misconfiguration, then a partner's data processing, and only its own investigation found attackers had compromised its internal credentials. And it still hadn't disclosed publicly as of the report.



Corn

Every draft of that apology was wrong until the last one.



Herman

Which is often how it goes. What I'd want Daniel to take from the whole picture is this: the Zero Data Retention label was doing almost nothing. Not a little — almost nothing. Two-thirds of ZDR-marked requests to one lab were accessed. Two of the three compromised providers had ZDR requests stolen. The setting is a contractual promise about the first hop, enforced by the party with the least visibility into what happens after it.



Corn

And we did an episode ages back on messaging apps — end-to-end encryption, and everything around it, metadata, backups, corporate ownership, that quietly determines your actual privacy. Same shape here. The guarantee covers the link, and everything that matters happens where the guarantee doesn't reach.



Herman

The marketplace economics are the deeper problem, though. A routing system that optimizes purely for price and performance, in a market where the thing being routed is worth more than the compute, will find the cheapest harvester every time. Not because anyone designed it that way — because nobody put a privacy term in the objective function. Moore's compression of it is the one I'd keep: "a privacy flag binds only the first hop."



Corn

There's also a reputational asymmetry worth naming. OpenRouter says the issue "sits with MiniMax and the providers that forwarded to it, not with OpenRouter." Defensible — and yet OpenRouter chose the providers, set the routing defaults, and issued the ZDR badge



Corn (continued)

those providers were wearing when the prompts leaked. "We weren't breached" and "your prompts were safe" are different sentences, and this week a lot of people discovered they'd been reading them as one.



Herman

They did also say "providers that want to be on OpenRouter will need to meet it" — the bar is going up. Whether the enforcement mechanism is anything more than self-certification plus spot checks is exactly the open question.



Hilbert

"Ordinary logging." You two keep saying that like logging is some small thing that got misused. The ordinary logs are the ones with everything in them. Nobody ever looks at the logs until the day everyone looks at the logs.



Herman

That's true in the boringest possible way, which is the worst way for it to be true.



Hilbert

I'll give you the one nobody counts. The URL field. A request log has the timestamp, the status code, the token counts, and the URL — and the URL is where the whole prompt sits, base-sixty-four and smiling, because the clients that put everything in the path were built by people who never imagined a log. Herman, you know this, you spent ten minutes on routing weights earlier and never once said where the request body actually gets written down. The field that was supposed to hold "endpoint" holds the customer's entire message, and the log rotation keeps it for ninety days, and the dashboard renders it. That's your marketplace seller. He didn't enable anything sinister. He enabled nothing. It was already on.



Corn

So the scariest actor in the report needed a checkbox, and didn't even need that.



Hilbert

I traced one once. A single request, mine, through two routers and a reseller, because I wanted to know how many hops a thing of mine had actually taken. I kept the paperwork on the desk for six weeks. My brother-in-law has it now, the whole trace, printed, and he won't give it back because he thinks it's evidence of something. The surprising result was the middle router. It sent my request to a provider I had never contacted, on a different continent, and when I wrote to ask why, the answer came back in the wrong tense. "This has been corrected." I never asked what it was before it was corrected. Four hops it took, not three. I counted the latency by hand.



Herman

Four hops for a request you thought had one.



Hilbert

The claw doesn't help with the printouts, but I manage. If Dragos is reading this — the seller adding a thousand hidden tokens, that's the one I'd watch. Tokens are money. Somebody is paying for those.



Corn

Let's pull back, because there's a lot still unresolved. Which three providers did OpenRouter off-board — Dragos never named its three breached providers either, and nobody's said whether the lists are even the same three.



Herman

Did MiniMax actually read the prompts, or did someone else — there's no answer, and MiniMax's only explanation went to a reseller, not to OpenRouter. What the "one exception" was that OpenRouter's email mentioned. Which Chinese lab Dragos means. And how much real customer data was copied — Dragos says the real requests processed alongside the bait were almost certainly copied too, but that's an inference. No provider has published a count.



Corn

If you take one thing from this episode, make it Moore's line: a privacy flag binds only the first hop. Everything after that first hop is trust, wearing a label.



Herman

The structural piece underneath it — the cheapest-provider incentive is still there, untouched. OpenRouter can raise the bar for who gets the badge, but until the routing itself can weigh whether a provider deserves the traffic, the discount remains the attack.



Corn

Watch whether MiniMax ever sends its own account. That silence is the tell.



Herman

Producer credit to Hilbert Flumingtop, who is presumably still counting latency by hand. If this was your kind of episode, go back for episode three, Safetensors or something else; episode sixteen, On Deepfakes, SynthID, And AI Watermarking; and episode thirty-five, The Privacy Gap. This has been My Weird Prompts.



Corn

If the episode earned it, leave us a review — it's how the weird questions find their way here. My weird prompts dot com has the archive. We'll be back soon.



Herman

See you tomorrow.

SOURCES

What the research for this episode read before the script was written. Primary sources first.

1. **OpenRouter, Data Processing Incidents on Inference Providers, 2026-11-22 [primary]**
(retrieved 2026-11-22)
https://openrouter.ai/blog/announcements/data-processing-incidents-on-inference-providers/

2. **Dragos (HD Moore), AI Supply Chain Security: What Canary Tokens Found, 2026-11-20 [primary]**
(retrieved 2026-11-22)
https://www.dragos.com/blog/ai-supply-chain-security-canary-tokens

3. **Novita AI, Privacy Notice Regarding MiniMax Requests on Novita, 2026-11-22 [primary]**
(retrieved 2026-11-22)
https://novita.ai/blog/minimax-requests-privacy-notice/

4. **OpenRouter docs, Provider Selection, undated [primary]**
(retrieved 2026-11-22)
https://openrouter.ai/docs/guides/routing/provider-selection

5. **The Hacker News, OpenRouter Drops 3 Providers After Researchers Catch Months of Prompt Theft at AI Hosting Firms, 2026-11**
(retrieved 2026-11-22)
https://thehackernews.com/2026/11/openrouter-drops-3-providers-after.html

6. **Ars Technica, A leaky AI supply chain is exposing secret user data at an alarming rate, 2026-11, updated 2026-11-21**
(retrieved 2026-11-22)
https://arstechnica.com/security/2026/11/a-leaky-ai-supply-chain-is-exposing-secret-user-data-at-an-alarming-rate/

7. **Privacy Affairs, OpenRouter: MiniMax Prompts May Have Been Read by Others, 2026-11-21**
(retrieved 2026-11-22)
https://www.privacyaffairs.com/openrouter-minimax-prompts/

8. **Hacker News, Tell HN: OpenRouter prompts to Minimax may have been read, item 50585962, 2026-11-21**
(retrieved 2026-11-22)
https://news.ycombinator.com/item?id=50585962

9. **Unite.AI, OpenRouter Cuts Three Providers After AI Prompt Exposure Findings, 2026-11**
(retrieved 2026-11-22)
https://www.unite.ai/openrouter-cuts-three-providers-after-ai-prompt-exposure-findings/

10. **Sulat AI & Tech News, Dragos canary study: three AI providers breached, ZDR ignored, 2026-11-20**
(retrieved 2026-11-22)
https://news.sulat.com/2026/11/20/dragos-ai-canary-prompts/