Scoop: Notion may have lied to customers, resulting in a data breach that impacted enterprise accounts

TL;DR

Notion told enterprise users that it would never store user data on AI providers' servers. However, Notion must have made some major internal mistakes, because it ended up storing a bunch of user data inside one of their AI providers, a company named "Fireworks."

Then, Fireworks got hacked, and a huge amount of Notion's user data got stolen by the hackers. If Notion had followed it's own policy and kept its promises to users, Notion users would not have been impacted by the Fireworks hack.

Context

Notion had a major data breach, which impacted many enterprise customers.

Notion uses Fireworks as an AI provider. Fireworks got hacked, and Notion user data was exposed in the hack. The Notion feature that exposed user data is called "Insights." If Notion had been following their own AI policy, enterprise user data would not have been exposed by this feature, and enterprise users would not have been affected by the Fireworks breach.

Notion has been telling their customers that Notion only uses AI providers who promise to never ever store Notion's data: the AI provider just processes the data, sends it back, and never saves it. That's called a "zero data retention" (ZDR) policy. ZDR is important, because if Fireworks stores zero data from Notion, and then Fireworks gets hacked, Notion's users are actually pretty safe. Fireworks doesn't have any data from Notion's users, so the hackers can't steal any of it. (Editor's note: I gloss over the nitty-gritty of what Notion told customers in this summary, but go into the details below.)

So, Notion customers thought that Fireworks wouldn't store any of their data. But they clearly did store some, because hackers were able to get a bunch of Notion user data from Fireworks's servers/storage.

Quick Notes
This is now finalized & live as a blog post and on my LinkedIn. (The full text is all here in this doc though)
If you are an enterprise Notion customer, I'd love to talk to you. I believe Notion violated their own AI policy, resulting in your data being exposed, and I'm digging into it. I have a handful of sources, but I'd love more eyes on this. You can reach me here (temporary email address so I can post it online without getting spammed), or DM me on LinkedIn.

Why was Fireworks storing Notion user data?

One of the Fireworks cofounders posted on Twitter/X claiming that Fireworks was storing Notion's user data because Notion chose to use the Fireworks Batch Inference API, and that that API is clearly documented as "not compatible with zero data retention."

We'll assume here that the Batch Inference API is the Fireworks endpoint that Notion used, and which collected stored data which was then later exposed.

It should be obvious to any engineer that a batch API is not zero data retention. The way a batch API works is, you send a bunch of data to the provider, they store that data and they slowly process it, and when they're done they send it back to you. Note that they have to store the data here!

So I have limited sympathy for Notion's claims that Fireworks was supposed to be using ZDR for Notion's data (more on those claims in the final section).

Did Notion violate its promise to users about ZDR?

Many customers seem to have been under the impression that their data was protected by ZDR, and they're very surprised that a third-party AI provider had been storing their data.

In fact, before this breach, Notion had a clear policy on ZDR. Here's an archive.org link to the relevant section, and a snippet of the text: “... there are certain AI-powered features that may require the use of data-retaining LLMs; in such cases, Notion will make available Workspace settings to enable administrators to turn on data-retaining LLMs. Otherwise, such data-retaining LLMs will remain off in your Workspace by default.“

So, Notion says that there may be some AI features that do not use ZDR, but also promises that those features are off by default on enterprise accounts, and can only be enabled if an account admin flips a switch to enable them.

Here’s the problem: I have had half a dozen enterprise Notion customers tell me that they have never turned on such a setting. In fact they can't find it anywhere in the admin console… and that they were automatically opted-in to "Insights," the AI feature that was non-ZDR and which resulted in a breach.

I spoke to several sources who are all account administrators on enterprise Notion accounts, and were affected by the breach.

One said Notion support has confirmed, in writing, that: "Insights was enabled by default (not opt-in), that on my account the option to decline it wasn't findable, and that while Fireworks was on the general subprocessor list, no separate notice was given about Insights itself." (Insights is the Notion feature from which data was exposed.)

Another source, understandably upset about the breach, said "My understanding is they enabled Insights by default. I certainly don't remember flipping a toggle, and I DEFINITELY didn't agree to forfeiting ZDR!!! There is no toggle currently." ("Toggle" here refers to the "Enforce ZDR" switch that Notion's AI policy says should've been available and defaulted to "ON".)

It certainly seems like Notion violated its own AI and ZDR policy, deceived (presumably by accident!) about the handling of their data, and as a result of their mishandling, exposed customer data to a breach.

Notion sent sensitive enterprise user data to a Fireworks endpoint that was clearly labelled as not compatible with ZDR. Further, it seems that they turned the “Insights” feature on for all users without any protection or opt-in, which is directly against their own policy.

In their existing communications, Notion has not acknowledged this. In fact, they seem to be attempting to gloss over it and push additional liability onto Fireworks. Of course, Fireworks had a breach and they're responsible for that — nothing in this article aims to absolve or reduce Fireworks's ownership of their own security.

How did this happen?

In short, I think this was a mistake that one of Notion's product/engineering teams must have made. This sort of mistake is more common than you might think. But then, we all got very unlucky because the mistake happened to actually lead directly to a major data breach.

When this sort of thing happens, it usually is a cascade of failures, where a broader protection fails, and then at a lower-level other checks also fail. So we can think about this in layers:

- **Contractual**: Notion signed a contract with Fireworks that had a "zero data retention" clause in it. (At least, that's what Notion claims: see the "I have ZDR as part of my Enterprise plan" section in the FAQ at the bottom of their security notice.) I think Notion may have assumed that the contract itself meant that Fireworks would automatically apply ZDR to all of Notion's data, and that Notion essentially wouldn't have to think about this issue — Fireworks would just never retain Notion's data no matter what.
- **Administration**: Fireworks has a setting that users like Notion can turn on, that will enforce ZDR for the whole account. If that setting were active, Notion would have been unable to use any Fireworks endpoints that require data retention. That includes the Batch Inference API, which is the Fireworks endpoint that backed Notion's "Insights" feature.
    - **Post-sales**: Presumably, someone at Fireworks should have activated the ZDR setting for Notion, or explained how it works. (And maybe they did!)
    - **Notion Admins**: Either that setting never got turned on, or it got turned off by a Notion administrator. The setting is "all or nothing."
- **Internal**: Then, internally, one of Notion's teams decided to build a Notion feature, "Insights," that uses the Fireworks Batch Inference API, and to opt-in all users to that Notion feature. That team may have misunderstood their internal policy on ZDR, just overlooked it, or they may have made a technical mistake, like intending to opt-in some user group but not all.

At the top layer, contractual, I do not know any details. However, it sounds like Notion may have thought that based on their contract, Notion wouldn't have to "think" about enforicng ZDR on their Fireworks instance. They thought that all of their Fireworks usage would be covered by ZDR, without any action or thought needed on Notion's part. And maybe Notion communicated that internally to their teams.

I can imagine an email being sent out along the lines of: "Good morning team! We're using a new AI provider, Fireworks. They're fully ZDR compliant, and cheaper than other providers, so we encourage you to move AI usage onto them. Thanks!" Now the whole team may think that they don't have to worry about ZDR with Fireworks. So engineers are operating under the assumption that all Fireworks usage is ZDR and they're just put the question of ZDR compliance out of their head.

At the administration layer, there is this "enforce ZDR" switch that you can turn on. Fireworks's post-sales team should have explained the switch to Notion's team, and then activated that switch for them, or made sure that Notion activated it. There could have been a fumble here. Maybe they forgot to explain it, maybe the Notion team member who they told about it forgot to turn it on, etc.

Then, even if the switch did get turned on initially, someone could have gone and turned it off. Maybe one Notion team wanted to test a Fireworks feature that retains data (aka is non-ZDR), and they promised they wouldn't put any user data in it during their testing. So one of Notion's admins turned off the enforce ZDR switch. But then maybe another team didn't realize that ZDR wasn't enforced anymore, and accidentally started using a non-ZDR feature for a real live, feature that sent use data to Fireworks.

Finally, at the internal policy, planning, and technical level, there must have been a mixup. It could've been that someone didn't understand that they were using a non-ZDR Fireworks endpoint. Or it could've been that they didn't understand Notion's own ZDR policy. Or, they could've accidentally opted enterprise users into the feature without intending to.

I think it's somewhat unlikely that engineers didn't realize that they were using a non-ZDR feature, because the Batch Inference API is obviously not ZDR compatible. Any engineer should have noticed that, as we discussed in the "Why was Fireworks storing Notion user data?" section. But maybe they just messed up and overlooked it. Maybe they were under intense time pressure and were just trying to get a feature out the door.

Or, the team didn't understand or consider Notion's own ZDR policy. This happens.

Last, but not least, and I actually think this happens often at many companies, it could've been a user test group issue. When you launch a new feature, you add different "user groups" to it, so that they can access it. It doesn't just turn on for everyone. So first you might add a user group that's "beta" users, aka accounts that have agreed to test new features for you. Then you might add "unpaid" users, and then "enterprise" users, etc. Often theses groups are poorly managed, or the interface to add them is prone to user error. So they may have accidentally added all enterprise users to the feature without intending to!

This is at least a 2-step cascading failure. The "enforce ZDR" switch should have been on, and stayed on. And the team building the feature should have known the policy and ensured that they complied with it.

You can see how these things are complex! I certainly feel sympathy for the people responsible for this at Notion. They messed up, of course, but I bet that they did not have the support and enforcement that they needed from the broader org.

Sources & Links

Notion:

- Security bulletin (Archive.org link, 2026-11-10)
- Notice about Fireworks incident (Archive.org link, 2026-11-10). This is the most detailed document they've shared. I do not understand why this isn't linked from the bulletin.
- Notion's AI Security Practices Policy:
    - Live link
    - Prior to the breach, on 2026-09-05 (Archive.org link). This is presumably the version that was applicable prior to and during the breach.

Fireworks:

- Twitter/X post by a Fireworks co-founder objecting to Notion's claims.
- Fireworks security bulletin
- Fireworks ZDR documentation

Other:

- A Reddit thread with Notion enterprise users complaining