

Lesson 13 — Incident Response for AI-Related Leaks

The leak has already happened. Somebody pasted something they should not have into a tool they should not have, and now your job is to make the next sixty minutes count. This lesson is the playbook for those sixty minutes.

Audience: Security + Leads

AI OpSec

Incident Response

Notes: The goal is calm, ordered action. The worst incidents are the ones where people improvise under panic and skip steps.

What you will learn

- The five-step playbook: **Contain → Assess → Rotate → Notify → Document**
 - The special case of the third-party training endpoint — what deletion really means
 - A minute-by-minute template for the first hour after a suspected AI leak
 - What never to do, even when it feels like the obvious move
-

The playbook at a glance

- **1. Contain** — stop the bleeding, freeze the evidence
- **2. Assess exposure** — what did the provider actually see?
- **3. Rotate** — credentials, tokens, sessions, in priority order
- **4. Notify** — internal first, then legal, then customers if required
- **5. Document** — blameless postmortem with a systemic fix

Notes: The order matters. People who rotate before they contain destroy their own evidence.

Step 1: Contain

- **Stop using the tool.** No more prompts, no more follow-ups, no "let me just check one more thing"
 - **Capture evidence** — screenshots of the prompt and response, the exact timestamp, the URL, the account identity
 - **Copy the prompt text verbatim** into secure notes. You will need the exact characters later
 - **Do NOT delete the chat.** The chat is evidence. Deleting it destroys your timeline and may still not remove it from the provider
 - Leave the tab open. Freeze the scene.
-

Step 2: Assess exposure

- **What did the provider see?** Not just the text — also the metadata: account, IP, timestamps, browser fingerprint
 - **Retention window** — how long does this provider keep prompts? Thirty days? Indefinitely? On this plan?
 - **Sub-processors** — who else gets the data? Hosting, logging, abuse review, training pipelines?
 - **Who else could access it?** Shared account, team workspace, chat history sync to another device?
 - Assume everything you sent is now in somebody else's database. Plan from there.
-

Step 3: Rotate

- **Priority order:** session tokens → API keys → passwords → long-lived credentials
 - **Session cookies first** — they are already active and cannot be un-sent
 - **API keys** — revoke, issue new, update every deployment that used them
 - **Passwords** — force rotation on any account whose password was in the chat context
 - **Long-lived secrets** — DB strings, signing keys, cloud IAM access keys, service account JSONs
 - Rotate everything the user touched in that session, not just the one thing they pasted. The blast radius is the whole session.
-

Step 4: Notify

- **Internal security** — first call, within minutes, not hours
 - **Management** — the lead who owns the affected system; they need to know before a customer does
 - **Legal / privacy** — if PII was in the chat, or if regulatory reporting clocks have started
 - **Customers** — if contract or law requires it, within the SLA stated there
 - Late notification is its own incident. Document when you notified, whom, and what you said.
-

Step 5: Document

- **Blameless postmortem** — the person who pasted the secret is not the root cause
 - **Root cause is almost always systemic** — the tool was approved, the training against using it for that kind of data was not
 - **Timeline doc** — what happened, when, who acted, what was rotated, what was notified
 - **Systemic fix** — DLP rule, approved-tool list, training update, redacting proxy, policy change
 - If the postmortem ends with "we told the engineer to be more careful," you have not finished the postmortem.
-

The "I pasted X into Y" special case

- The third-party training endpoint is the most common AI-specific incident: consumer chat account, corporate data pasted in
 - **Request deletion under the provider's terms** — most providers have a form, some honour it, none prove it
 - **Deletion is a contractual courtesy, not a technical guarantee.** Your incident plan must assume the data is gone for good
 - If the account was personal (not corporate), you also have no paper trail of the request you sent
 - Treat the provider side as best-effort. Do all your real work on the rotation and notification side.
-

First-hour timeline template

- **T+0** — acknowledge, tell reporter to freeze the tab, capture screenshots, do not delete
 - **T+2** — identify exactly which secret, which system, which scope
 - **T+5** — rotate the primary credential
 - **T+8** — check audit logs for use of the old credential since the paste time
 - **T+12** — rotate everything else in the same chat session
 - **T+20** — notify security lead and the owning manager
 - **T+30** — start the timeline doc, assign a postmortem owner
 - **T+45** — customer / legal notification if required
 - **T+60** — submit provider deletion request, file the incident ticket
-

What NOT to do

- **Don't delete the chat.** You are destroying your own evidence, and it probably will not delete it on the provider side anyway
 - **Don't wait** "until you're sure it was real." The rotation cost is cheap; the delay cost is not
 - **Don't hide it** from management or security. Hidden incidents become career-ending incidents
 - **Don't rotate only "the obvious" credentials.** The blast radius is the whole chat session, not the one line the engineer remembers pasting
 - **Don't blame the reporter.** A culture that punishes reporters gets fewer reports, not fewer leaks
-

Rules to remember

- Contain first. Capture evidence before anything else. Do not delete the chat.
 - Assume the provider saw everything you sent, including metadata, including sub-processors.
 - Rotate priority: session tokens > API keys > passwords > long-lived creds. All of them, not just the "obvious" ones.
 - Notify within your documented SLA. Late notification is its own incident.
 - The postmortem is about systems, not people.
 - Third-party deletion requests are a contractual courtesy, not a technical guarantee.
-

