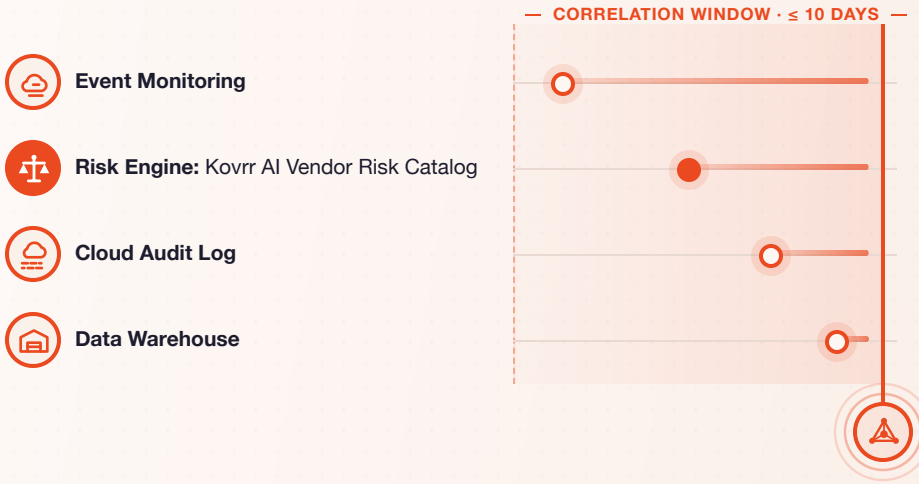
KOVRR AI INTERACTION DATA FABRIC IN ACTION

A Teardown of the Drift OAuth Supply-Chain Compromise · What the Customer-Side Logs Could Show

Three Minutes, Fourteen Days



Four days of quiet API work, then on 14 August three object counts inside thirteen seconds, Account, Contact, User, followed by broad Case queries. The reconnaissance, from two ordinary hosting addresses the vendor does not run. The trigger fires on the counting.

Resolves the connected app to the vendor and, from each estate's own inventory, every other grant that vendor holds. Runs after the trigger and scopes the response, so revocation reaches the vendor rather than one visible app.

Where a harvested access key would be used if it were used. First appearance of a recovered credential against infrastructure the vendor does not operate. The risk the export creates, not something observed in this campaign.

The same downstream risk in the data platform, an unfamiliar principal scanning volume above its own baseline. Modelled from the exposure, not seen in this incident.

Scoped Day One, Not Week Three

● KOVRR NATIVE ○ THIRD-PARTY TELEMETRY

RISK SCENARIO EXPLAINED

In August 2025 an attacker used the standing OAuth tokens a third-party sales-chat vendor held for its customers and exported records from those customers' Salesforce systems. Google's threat-intelligence team dates the activity from as early as 8 August through at least 18 August, a lower bound at both ends. The target was the free text, years of support tickets in which someone had pasted a key or a password to resolve an issue faster. One affected company searched the stolen data afterward, found 104 of its own API tokens inside it, and rotated all 104. The exposure was never the contact list. A supplier's breach handed someone a directory of credentials, and no one could say which ones without reading years of tickets.

THE INTERVAL, NOT THE THEFT, IS WHERE THE COST LANDS

THE EXPORT Just over three minutes, the company's own measurement

The interval

THE FIRST NOTICE Fourteen days after the first trace of the actor in the company's own logs

The theft was fast and the record of it survived, since the deleted export job left residual logs a post-mortem could reconstruct. What stretched was the interval before anyone knew. Notification clocks start at awareness, and awareness arrived two weeks after the first trace of this actor appeared in the company's own logs, a failed check of a recovered token against that company's own API on 9 August. Measured from the export itself the delay is six days. Either way the exposure ran before the clock started, and materiality then had to be judged without knowing what the records held.

EACH LOG OWNS ONE LINK. CORRELATION SPANS THE CHAIN.

[redacted]
1 · Vendor code repository
Compromised March to June 2025. No customer log contains this.

[redacted]
2 · Vendor cloud, token theft
The OAuth tokens taken from the vendor's environment. No customer log contains this.

CUSTOMER VISIBILITY BEGINS HERE

TRIGGER
3 · Enumeration
Counts and describes and broad queries against corporate Salesforce. Fires on the counting, not the export.

EXPORT
4 · The export
A three-minute bulk export, job deleted. Probably absent from the trigger log, which is why the rule is built on stage 3.

DOWNSTREAM RISK
5 · Downstream use
A harvested credential exercised in AWS or the Data Warehouse. The risk the export creates, not observed in this campaign.

SCOPE, ON TRIGGER Once the trigger fires, the vendor catalog scopes the response to every grant the vendor holds, so revocation reaches the vendor rather than the one visible app.

No single log spans the chain, and the first two links sit in no customer log at all. Correlation across the customer-side logs turns a supplier's disclosure into a dated, scoped finding rather than a two-week wait.

ALSO INTEGRATED IN THE FABRIC, NOT TRIGGERED THIS EVENT: IDENTITY · BROWSER · ENDPOINT · DLP

WHY SIGNAL TRIANGULATION WINS · EACH SOURCE'S BLIND SPOT IS ANOTHER'S STRONG SUIT

- **Event Monitoring** caught the reconnaissance, three object counts in thirteen seconds from an address the vendor does not run, with no reliable record of the three-minute export that followed.
- **AI Vendor Risk Catalog** resolved the connected app to its vendor and every other grant that vendor held, with no view of the traffic itself, so it scopes the response rather than raising the alert.
- The **Cloud Audit Log** would carry the first use of a harvested key against infrastructure the company does not operate, with no link back to the export except the credential string itself.
- The **Data Warehouse** would carry the same downstream use in the data platform, an unfamiliar principal above its own baseline, modelled from the exposure rather than seen in this incident.

Only correlation across the customer-side logs establishes that one vendor's stolen grant enumerated a corporate CRM, that the export carried free text full of credentials, and that any of those credentials could surface later in a cloud the company does not run. A supplier's disclosure becomes a scoped finding on day one rather than a rotate-everything guess in week three.

How the Correlation Is Built

The chain has four customer-side logs and two invisible steps. The first two, the vendor's code-repository compromise and the token theft from the vendor's cloud, appear in no log a customer owns, and no honest teardown draws a detection for them. Everything from the enumeration onward runs through logs the customer can own, though owning them is a purchase rather than a default.

START WITH WHAT THE CUSTOMER CANNOT SEE

The actor held access to the vendor's code repository from March to June 2025 and reached the vendor's cloud environment to take the OAuth tokens the vendor held for its customers. No customer log records either step. From the point those tokens were exercised against corporate Salesforce, the activity runs through customer-visible telemetry, with one large caveat. That telemetry is not owned by default. The trigger log requires the Salesforce Shield or Event Monitoring add-on and a specific user permission, a subscription rather than a setting, and the most likely reason none of this runs in a given tenant today.

THE CAPTURE SURFACE IS UNRESOLVED, TEST IT FIRST

One contradiction governs the whole first leg. The event-monitoring object documents its capture surface as SOAP and Bulk API for the Enterprise and Partner WSDLs, which names neither REST nor Bulk API 2.0. The incident traffic was REST, and the object's own type field documents a REST value, so the documentation contradicts itself. This did not resolve from published docs. Confirm that event rows appear for REST query and count calls in a live tenant before building anything on this leg. Everything downstream of the trigger assumes an answer not yet in hand.

THE FOUR SOURCES AND THE JOIN KEYS

Three join keys are exact and one is honest about its weakness. Within a session the login key ties events together, but the published victim timeline shows five separate login sessions across five days, so a rule keyed on session identity yields five unrelated incidents. Only the connected-app identifier spans them, and it is the single field that makes the detection possible. The connected app resolves to the vendor through the catalog. The fourth key is the weak one. Nothing links the CRM export to any later credential use except the credential string itself, which the company possesses only if it scanned the exported records for secrets. The affected company reported building custom scanning tools, regex and entropy and pattern matching, the same class of tool the actor was already running over the same records. Only one side had it before the incident.

RULE LOGIC

Plain text. Confirmed in the vendor's published schema.

Marked provisional. Vendor documents the capability but the exact key is unconfirmed.

Abstracted to plain language. Kovrr-native and not yet a shipped schema.

```
R1 · TRIGGER  (Salesforce Event Monitoring)         fires the incident
   G  = a third-party integration's connected-app ID
   B  = the org's own [redacted]-day history for G          // your baseline; the catalog has none

   PROVENANCE
     SourceIp NOT IN B.source_ips
        // or at ASN level, an enrichment you compute on SourceIp, not a Salesforce field
     OR UserAgent NOT IN B.user_agents
        // provisional: UserAgent is documented as "platform or environment," not the HTTP header
   BREADTH
     distinct QueriedEntities for G in 24h  >  [redacted]
     OR QueriedEntities touch objects G has never queried
   ESCALATOR (OR, never a precondition)
     QueriedEntities include free-text objects (Case, Task, EmailMessage, Note)
     OR RowsProcessed > [redacted]

   FIRE ON provenance AND breadth.   Escalator raises severity only.

R2 · SCOPE  (Kovrr AI Vendor Risk Catalog)          runs after R1, adds nothing to it
   connected-app ID -> vendor -> every grant that vendor holds,
   assembled from each estate's own inventory
   emit the revoke list keyed on the vendor, not the app

R3 · DOWNSTREAM  (CloudTrail | Snowflake)           window = credential lifetime
   for each secret recovered by scanning the exported records
     CloudTrail  where accessKeyId = <key>  OR arn = <arn resolved from that key>
                 and the source ASN is not one the company operates
     Snowflake   where the principal is the harvested user
                 and bytes scanned > [redacted] for that principal
```

Why the escalator is an OR and not an AND, the mistake worth internalizing. Every event the trigger log plausibly captured in this incident was a count or a small limited query, and a count returns one row. The single high-volume event, the export, is the event the log may not capture at all. A rule that requires a row-volume threshold alongside the provenance test therefore fires on nothing in this incident. Volume is an escalator here, not evidence.

WHY THE WINDOW IS DAYS

The bound is credential lifetime, not log latency. A long-lived access key sitting in a support-case body stays valid until someone rotates it, so export-to-use is unbounded until rotation, and the ten days is the documented campaign span rather than a ceiling. Log latency runs comfortably inside that, warehouse history within roughly 45 minutes to three hours, so a days-wide window absorbs it easily. A minutes-wide correlation window would have held a dozen disconnected events across six days and joined none of them.

WHAT ACTUALLY CLOSES IT

Revoke on the vendor, not the visible app, since the same vendor's other grants stay live otherwise, which is exactly what the threat-intelligence update warned. Scan the exported record set, rotate what it contains, and hunt first use of those specific credentials across every estate. Then close the setting that was open all along. Connected-app policies and login-IP ranges let an administrator constrain where a grant may be used, and almost nobody sets them for an integration whose egress the vendor can change without notice. A configuration left unset, not a missing product.