Post-Mortem: Incident of May 29, 2026

Incident Summary

On Friday, May 29, 2026, between 11:14 and 11:25 am UTC+2 (a span of 11 minutes), a configuration error on our content delivery network (CDN) caused our API responses to be cached unintentionally.

As a result, some users were able to see on their screen information belonging to other users within their organization.

The incident was detected internally within a few minutes, the configuration was fixed at 11:21 am, and normal operation was restored on the server side by 11:25 am. Given the nature of the data involved (personal data), we immediately activated our crisis team, then in a second step, (14 hours after the incident) individually notified by email each of the affected clients, sharing the associated logs so that each client could carry out its own analysis of the data involved.

One factor, however, may have extended the effects of the incident beyond those 11 minutes: the users' browser cache *(see the Impact section)*.

Timeline of events on May 29, 2026

- **11:14 am:** Lucca deployed a change to the cache configuration on our CDN. This change produced the opposite of the intended effect: instead of preventing certain responses from being cached, it enabled the caching of API responses.
- **11:15 am:** Caching of responses effectively began.
- **11:19 am:** First report and opening of the incident by our teams.
- **11:21 am:** Cause identified, rule disabled, and cache purged.
- **11:22-11:24 am:** Deployment of the corrected configuration (cache rule disabled), and classification as a security incident: End of cache effects on our platform.
- **11:27 am:** Crisis team activated.
- **Afternoon:**
    - Fixes to force the clearing of caches within browsers.
    - Log extraction and precise assessment of the impacted clients.
- **3:25 pm :** End of residual effects in the end users' browser caches (on disk).
- **End of day:**
    - Correlation of CDN logs with the IPs and logged in users.
    - Consolidation of the descriptions and criticality level of the routes.
    - Verification of the accuracy of the logs.

- **May 30, 12:55 am - 1:04 am:** Individual notification of all affected clients by email, including detailed logs of the impacted routes by criticality and of the users involved.

Impact

- **Nature of the exposure:** The incident was limited to an **unauthorized display** of data, in **read only** mode. **No modification, deletion, or export of data was possible.** A user could not choose the information they saw: they saw whatever the cache served them, with no way to target a specific person.
- **How the exposure worked:** For a given page, the information of the **first user** who viewed that page after the cache was activated was stored, then **shown to the following users** viewing the same page. Since our CDN cache is indexed **by request and not by user**, an employee could thus see displayed, on their own screen, the name or information of **another employee in their organization**.
- **The factor that extended the impact through the browser cache:** The main effect lasted 11 minutes on the server side. Only users who viewed pages during this window were affected by this extension: the data displayed at that time (for example the name shown in the application's banner) may have remained stored locally in their browser cache for up to 4 hours. These residual effects disappeared at 3:25 pm.
- **Data involved:** The exposed data was personal in nature: identity information (last name, first name, email) and, depending on the screens viewed, broader HR information (entity, absence balances, contracts). In **a more limited number of cases**, more “sensitive” data may have been displayed (absence supporting documents, payroll related items). All affected clients were contacted and received a **detailed and personalized log analysis**.
- **Scope of the exposure** : The scope was limited to the organization's data. No one outside the organization could access the data.

Root Cause

The main cause is a **human error in the cache configuration** on our CDN (CloudFlare). A change meant to prevent the caching of error responses (404s) had, as a side effect, enabled the caching of our API responses (the opposite of the original intent). This change had previously been deployed to pre-production (and on a single domain) without surfacing the error.

Two organizational factors allowed this error to reach production:

- **A flawed review:** on what was nonetheless a critical change, the modification was approved by a single person (other than the one proposing the change) with an insufficient level of seniority and this review raised no objection.

- **Inadequate testing:** The testing procedure was not carried out with a method able to detect a caching defect (multi-user verification, or checking the cache headers on API calls).

Our approach following this incident of May 29, 2026 therefore aims to put in place several layers of controls that are independent, so that no isolated error can ever again cause this type of incident.

Corrective and Preventive Actions

Below we detail our action plan, which was designed to multiply control points and reduce risk.

Actions already completed:

1. Strengthened control over cache changes: any change to the cache configuration now requires the **approval of 4 named and qualified individuals**, before any change can be validated. No cache rule change can any longer be approved by a single person or without the sign off of these 4 named reviewers. This short term protective measure locks down the risk.

Actions in progress:

2. Protection at the source: we will add explicit “do not cache” directives directly on our APIs (via headers), **on the server side**. This additional measure, independent of the CDN, would greatly reduce the risk of API responses being cached in the event of a future configuration error.

3. Automated testing: setting up automated tests that fail if an API response gets cached.

4. Automated detection: creation of an alert triggered by any abnormal caching behavior on our APIs, with automatic incident triggering.

5. Ability to revoke browser and mobile caches : the ability, in the event of an incident, to log out all affected users and reliably purge the browser and mobile caches.

6. Gradual rollout of configuration changes: setting up a batched deployment process (with a finer granularity than pre-production / production), so that a single change cannot impact all of our clients at once.

Actions under study, ETA early 2027:

7. Reducing our reliance on caching: we are reassessing the value of caching and the solution used for our CDN cache to determine whether this usage can be reduced, or even removed entirely. Replacing our current solution (CloudFlare) with a WAF only solution (without a CDN) would completely eliminate the risk of a cache rule incident.

8. Emergency contacts in the Client Area: as announced in 2025, we are continuing to improve the reliability of the **Security and DPO contacts** for your organization, so that we can reach you **as quickly as possible** in the event of an incident. We encourage you to check and keep this essential information up to date. To add or update these Security and DPO contacts, see our help article available here.

Conclusion

We wish to offer you our most sincere apologies for this incident. Please be assured that the security of your data and your trust are our priority.

The action plan we are sharing with you aims to minimize the risk tied to the use of the caching mechanism in the short term and to explore the structural changes that would allow us to remove it in the medium term.

To be informed in real time of any incident or maintenance operation, we also invite you to subscribe to our Status page: https://status.lucca.fr

We remain fully at your disposal for any questions, through your usual point of contact or our security and DPO contacts: security@luccasoftware.com and dpo@luccasoftware.com

Kind regards,

Nantes, June 11, 2026

Raphaël DUCOM
Chief Information Security Officer