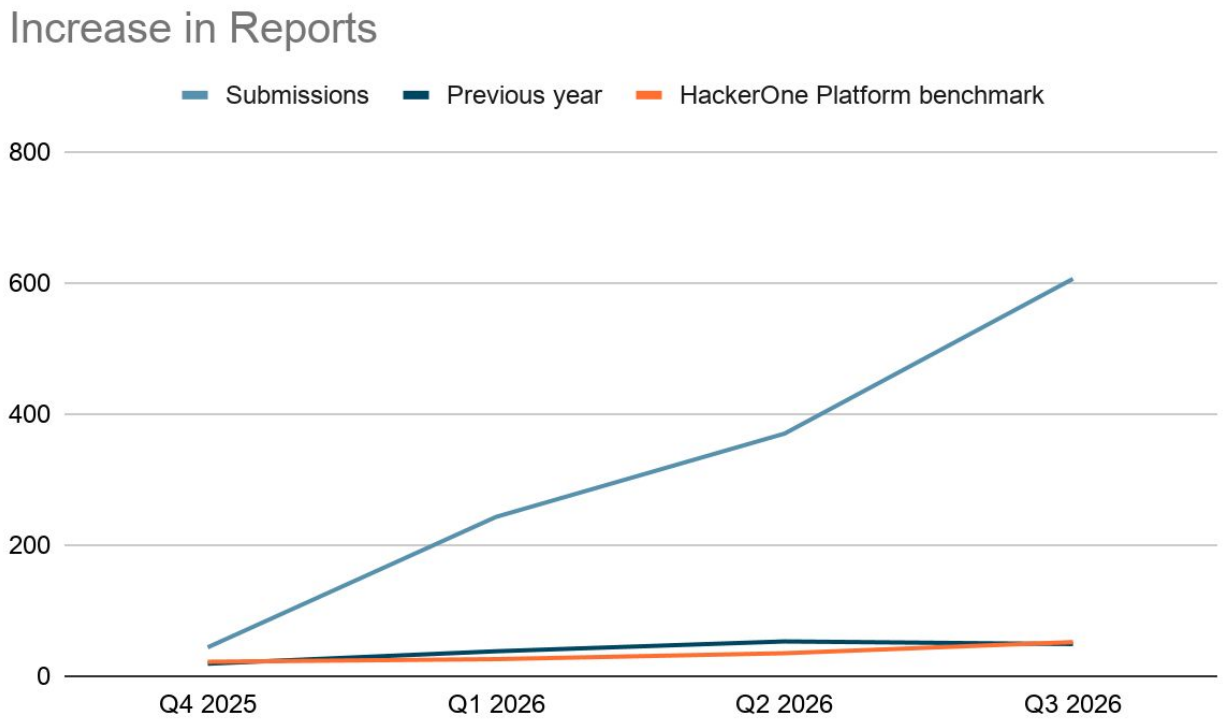
The Future of SRC

Mo Khan & Standa Láznička, Microsoft

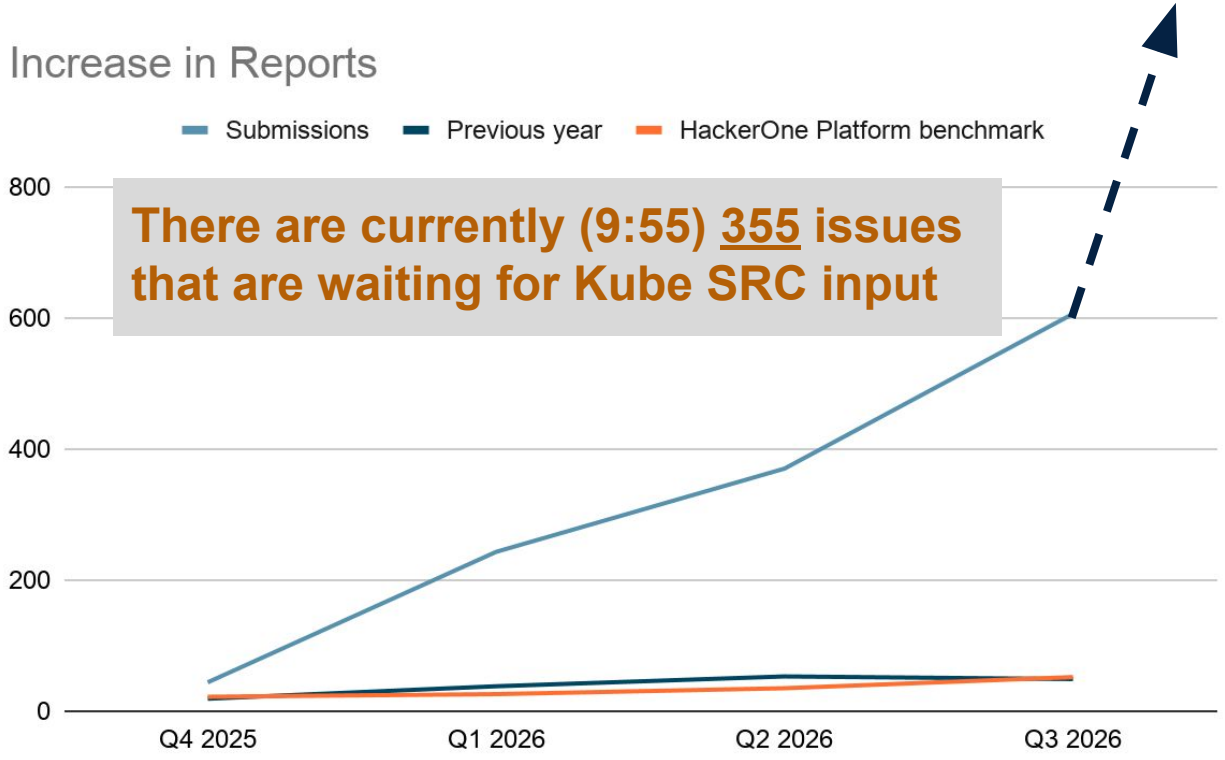



NO
MORE
CURL AND LIBCURL
THE END OF THE CURL BUG-BOUNTY
JANUARY 26, 2026
DANIEL STENBERG
2 COMMENTS

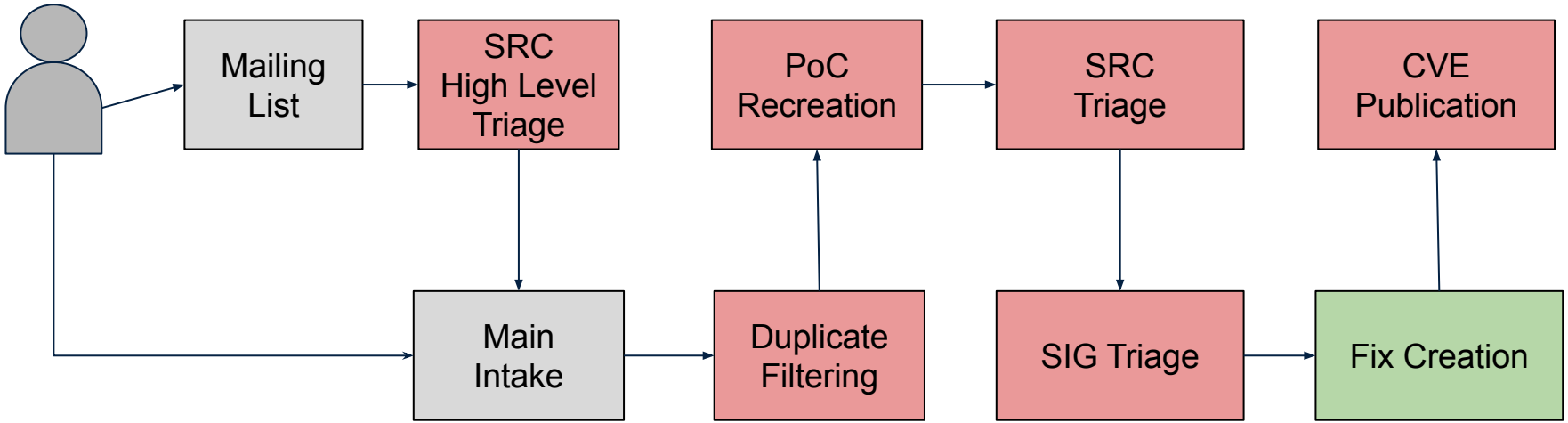
The Problem!


Increase in Reports
Submissions
Previous year
HackerOne Platform benchmark
800
600
400
200
0
Q4 2025
Q1 2026
Q2 2026
Q3 2026

The Problem!


Increase in Reports
Submissions
Previous year
HackerOne Platform benchmark
800
600
400
200
0
Q4 2025
Q1 2026
Q2 2026
Q3 2026
There are currently (9:55) 355 issues that are waiting for Kube SRC input

Typical Security Response Process



Proposed Changes

1. Automatic copying of H1 data into private repo
2. All SIG leads have access
3. Auto AI labeling for SIG, severity
4. Auto notification for SIG leads
5. Minimal tracking info in k/k with SIG labels
6. For low/medium issues, auto disclosure in 4 weeks unless opted out

h1-4171815: kube-apiserver: anonymous auth declared via --authentication-config bypasses the AlwaysAllow check #113

New issue

Open

enj opened 7m ago — Last edited by enj

`kube-apiserver` 's startup interlock (`pkg/kubeapiserver/options/authentication.go` , `ApplyAuthorization`) exists to force anonymous authentication off whenever the authorizer defaults to `AlwaysAllow` , and it logs `"AnonymousAuth is not allowed with the AlwaysAllow authorizer. Resetting AnonymousAuth to false"` when it does so. The interlock only ever mutates the legacy `--anonymous-auth` flag's in-memory struct field; when anonymous auth is instead declared inside the structured file passed via `--authentication-config` , `ToAuthenticationConfig` later reads that file's `anonymous.enabled: true` value directly and builds the real authenticator from it, bypassing the reset entirely while the startup log still claims success. On a server started with such a file and no `--authorization-mode` / `--authorization-config` (which defaults to `AlwaysAllow`), every unauthenticated request is accepted as `system:anonymous` and authorized as if cluster-admin. We independently verified the cited source against live `kubernetes/kubernetes` `master` (exact file:line matches for both the interlock and the config-precedence logic) and fully reproduced the exploit end-to-end against the official `v1.37.0` release binary with a real `etcd` backing store.

Steps to Reproduce

Reproduced on upstream `kube-apiserver v1.37.0` (official release binary, SHA-256 verified against `dl.k8s.io`), running standalone against a real `etcd v3.5.21` , no `kubeadm` / `minikube` / `kind` involved since the finding is purely an API-server authn/authz defect.

1. Create the `AuthenticationConfiguration` file:

```
apiVersion: apiserver.config.k8s.io/v1
kind: AuthenticationConfiguration
anonymous:
  enabled: true
```

2. Start `kube-apiserver` with `--authentication-config` pointing at that file, and do NOT pass `--anonymous-auth` , `--authorization-mode` , or `--authorization-config` :

Assignees
No one - Assign yourself
Assign to Agent

Labels
auto-disclosure | h1-triaged | severity/low | sig/auth | triage-accepted

Type
No type

Fields
Priority — Choose an option

Projects
No projects

Milestone
No milestone

Relationships
None yet

Working Together: Faster Triage, Less Noise (01 / 05)

Automation assigns initial SIG & severity and brings in SRC and SIG Leads in parallel.

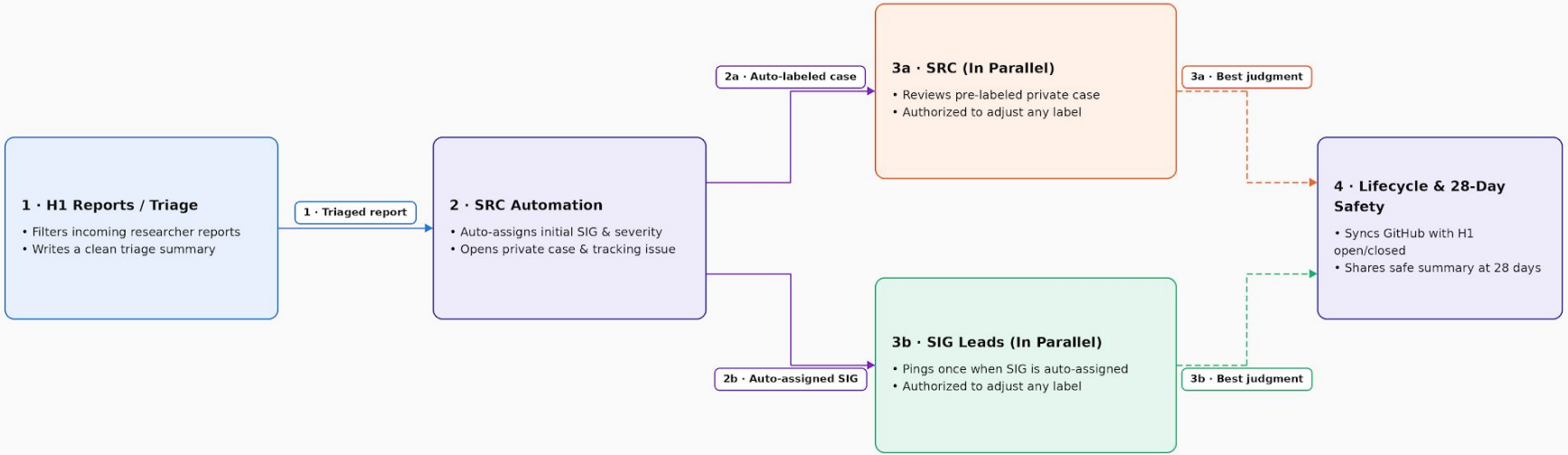
H1 Reports / Triage · Validates & summarizes reports

SRC · Co-reviews & adjusts severity/routing

SIG Leads · Co-reviews & adjusts labels/fixes

SRC Automation · Auto-assigns SIG, severity & sync

End-to-End Flow — Automated SIG & severity routing with parallel SRC + SIG Lead review



Why This Helps Everyone

Automation assigns initial SIG & severity · **SRC & SIG Leads review in parallel** · **Both can make best-judgment calls**

Intake & Auto-Routing: Ready for Parallel Review (02 / 05)

Automation imports the case, applies initial SIG & severity labels, and opens it to SRC & SIG Leads at once.

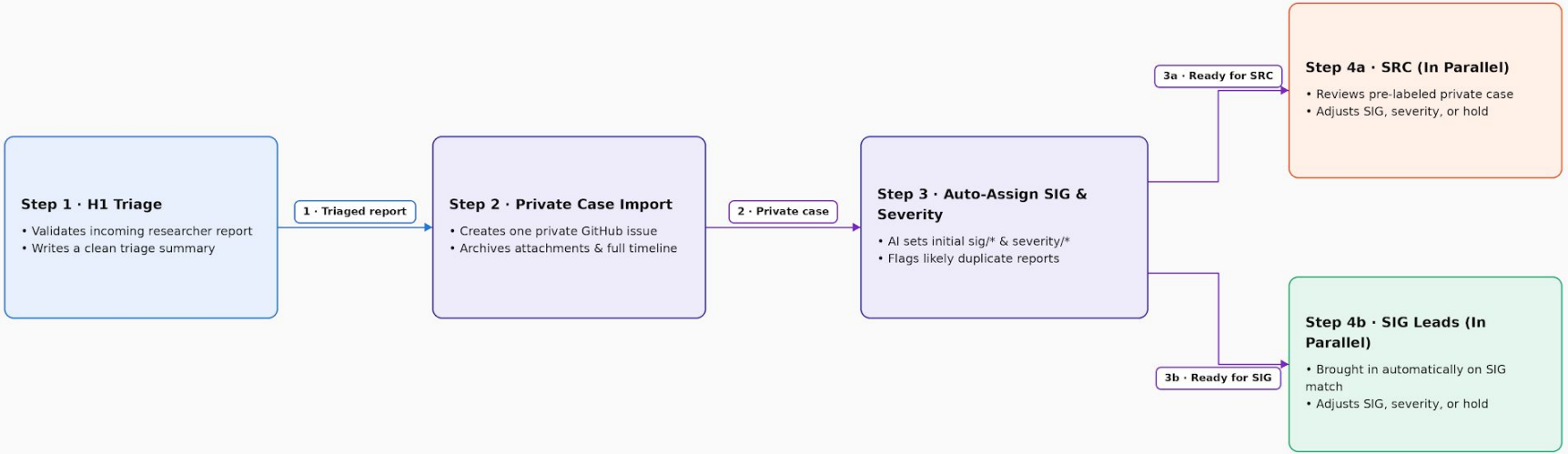
H1 Reports / Triage · Validates & summarizes reports

SRC · Co-reviews & adjusts severity/routing

SIG Leads · Co-reviews & adjusts labels/fixes

SRC Automation · Auto-assigns SIG, severity & sync

Steps 1 → 4 — Automated initial SIG & severity assignment, then parallel human review



Shared Human Authority
Automation assigns initial SIG & severity · **SRC & SIG Leads both make judgment calls** · **Human edits are never overwritten**

Parallel Review & Quiet Notifications (03 / 05)

Automated SIG assignment immediately notifies SIG Leads while SRC and SIG Leads collaborate as peers.

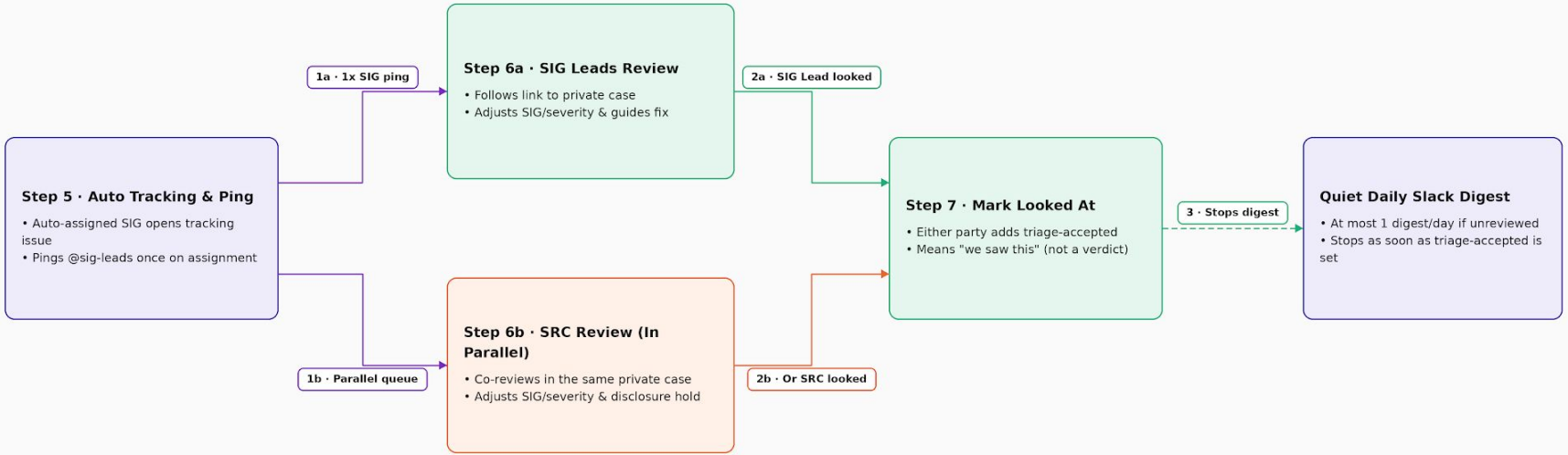
H1 Reports / Triage · Validates & summarizes reports

SRC · Co-reviews & adjusts severity/routing

SIG Leads · Co-reviews & adjusts labels/fixes

SRC Automation · Auto-assigns SIG, severity & sync

Steps 5 → 7 — Automatic tracking & ping, parallel co-review, and one-click reminder stop



Equal Authority, Zero Nagging
Automation triggers the initial SIG ping · **Both SIG Leads & SRC can edit labels** · **One click stops Slack reminders**

GitHub Guide for SIG Leads: Two Places, Clear Roles (04 / 05)

Neutral tracking stays on your public board, while all real details and label controls live in one private issue.

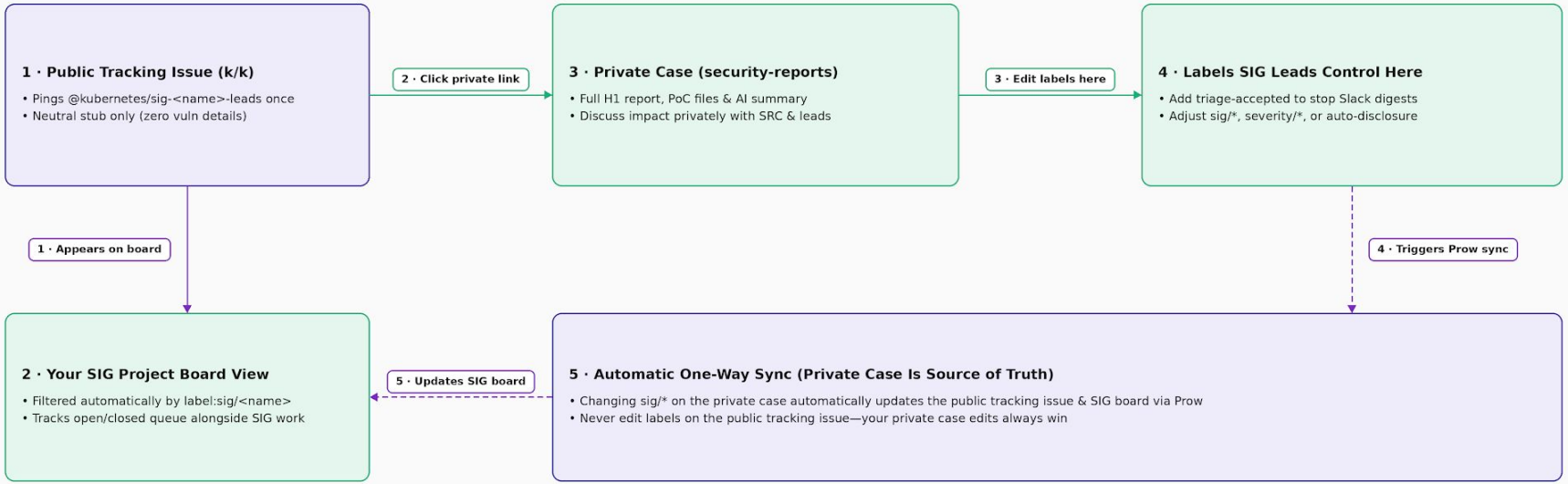
H1 Reports / Triage · Validates & summarizes reports

SRC · Co-reviews & adjusts severity/routing

SIG Leads · Co-reviews & adjusts labels/fixes

SRC Automation · Auto-assigns SIG, severity & sync

Where SIG Leads Work — Public tracking for board visibility vs. private case repo for full details & labels



What SIG Leads Need to Remember

Click the link in the tracking issue · **Read & edit labels** only in the private repo · **Public board updates itself** automatically

Safe 28-Day Disclosure & Automatic Cleanup (05 / 05)

Routine low/medium summaries share transparently at 28 days, while SRC or SIG Leads can hold any case.

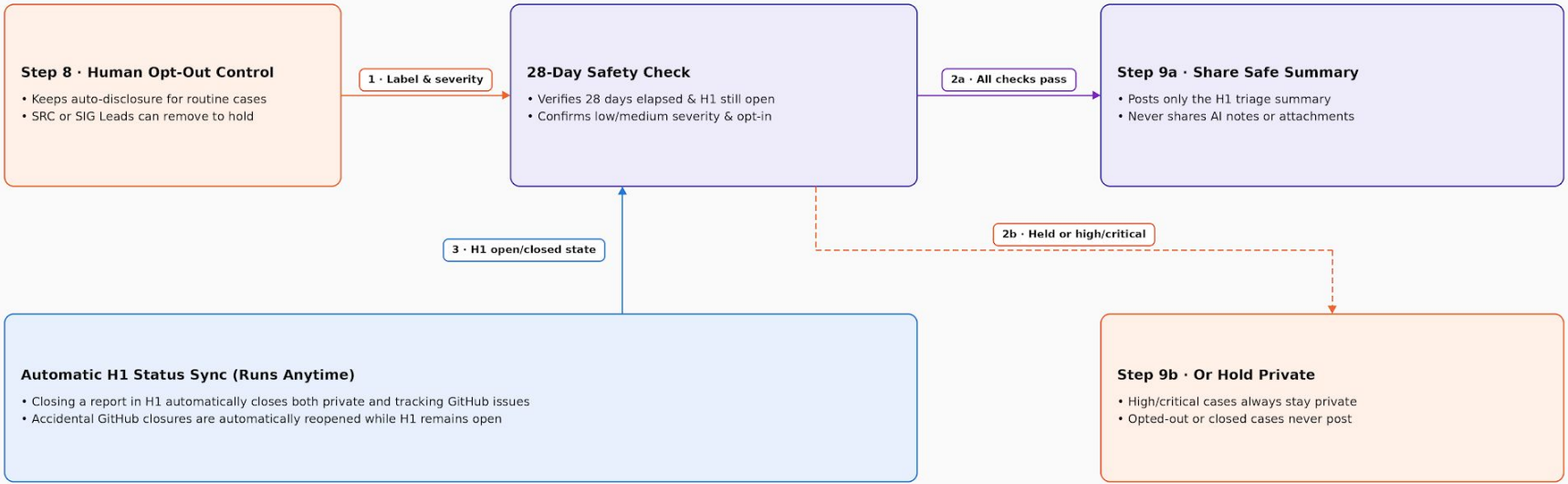
H1 Reports / Triage · Validates & summarizes reports

SRC · Co-reviews & adjusts severity/routing

SIG Leads · Co-reviews & adjusts labels/fixes

SRC Automation · Auto-assigns SIG, severity & sync

Steps 8 → 9 — Predictable 28-day sharing with human opt-out and automatic H1 lifecycle sync



Peace of Mind by Default
High & critical cases never auto-disclose · **One label removal** holds any case private · **Closing in H1** automatically cleans up GitHub

Community feedback

- What do you think?
  - Yay!
  - Nay!