



Don't Merge That

Joris Bonnefoy

Staff Software Engineer, Datadog

#58204	Bump client library to 4.2	MERGED
#58207	Add retry to export job	MERGED
#58211	Raise pool size for ingest	MERGED
#58213	Gate EU export behind rollout flag	HIGH RISK
#58216	Fix typo in dashboard title	MERGED
#58219	Tune cache TTL for query nodes	MERGED

An Agentic Approach to Catching Outages at **10,000 PRs a Week**



Joris Bonnefoy

Staff Software Engineer, Datadog

ABOUT ME

○ 8 YEARS

working in SRE and Security at Datadog, focused on large-scale failure modes and disaster recovery

○ ~4.5 YEARS

managing the Chaos Engineering teams and building our fault injection mechanisms

● Now AI DEVX

building reliability and security policies into agentic workflows

More changes make reliability harder to govern.

THE CHALLENGE

Governing reliability at today's pace

Engineers and coding agents ship more changes, to more services, every day. Reliability turns into a patchwork: strong in some teams, thin in others, and uneven from one service to the next.

MORE CHANGES

+66%

~52,000 PRs merged a month
pull requests in two quarters

84% vs 30%

~780 vs ~280 of 933 engineers

say AI speeds up writing a PR
vs. reviewing one

THE RELIABILITY COST

1-2%

≈ 500-1,000 PRs a month
of pull requests cause incidents

~20%

≈ 50 workdays a year
under code freeze, blocking
4,000+ engineers

From one outage to how the whole company ships software

INCIDENTS

Observe and operate

How we grade, run and review incidents

RISK MITIGATION

Capture and remediate

Turning an outage into agentic rules and remediation campaigns

AGENT EVALUATION

Durable knowledge

Using incident history to evaluate and calibrate our agents

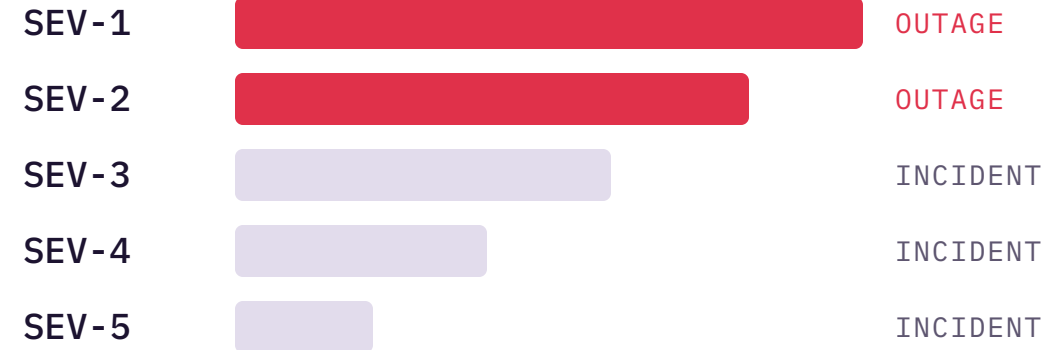
RISK-BASED REVIEW

Match review to risk

Scoring each change and matching the agentic review to its risk

Every incident gets a severity. SEV-1 and SEV-2 are outages.

SEV-1 is the highest severity and SEV-5 the lowest. Outages bring in a core incident commander to lead the response, in addition to the owning team.



Outages are collected, reviewed and followed through.

RESPOND Core IC A core incident commander leads every SEV-1 and SEV-2	COLLECT Outage record Outages are collected as they happen, with their timeline and context	WRITE Postmortem Required for every outage	REVIEW Quality review SRE teams review each postmortem for quality	FOLLOW UP Action items Engineering teams track follow-up items to completion
---	---	--	--	--

The result is a large, consistent history of outages. The rest of this talk is about what we do with it.

We use outages in two ways to improve resilience.

PATH 1

Risk mitigation

Find the general risks an outage exposed and reduce them across the organization.

PATH 2

Agentic training & evaluation

Use incident data to measure, calibrate and train the agents that review our changes.

The feature flag resolver went down. Services then failed one by one.

STEP 1: TRIGGER

Resolver outage

The feature flag resolver became unavailable.

STEP 2: FAILOVER

Fallback chain

2a **Cached value**
Services keep serving the last value they resolved

if not cached

2b **Default value**
The default set in code is used instead

STEP 3: AMPLIFIER

Cache loss

The cache is in memory. Services redeploy continuously, sometimes every minute, and each restart empties it, so 2a is skipped and every flag lands on 2b.

STEP 4: IMPACT

Stale defaults

Defaults set months ago no longer matched the live values applied everywhere. Services started failing.

One outage reveals a general risk. We write it as an invariant.

GENERAL RISK

Any feature flag whose default value no longer matches its live value

Not specific to the resolver, or to one service.
Any service reading flags is exposed.

becomes

INVARIANT reliability-pack / feature-flags

INV-3

A flag's default value keeps the service working when its live value can't be resolved.

A property that must always hold. Agents try to break it on every change, and every finding cites it.

Each rule ships inside a pack.

A pack is a short, versioned model of one system, written for review agents rather than for humans.

● ● ● reliability-pack / feature-flags	
SCOPE	Applies to: services that read feature flags at runtime. Out of scope: the flag service itself
INVARIANTS	INV-3 A flag's default value keeps the service working when its live value can't be resolved
REVIEW PRIORITIES	Flags with a live value set in all datacenters that differs from their default value

WHAT THE AGENT ASKS

"This change touches a service that reads feature flags, so INV-3 applies. Does any flag here have a live value in every datacenter that differs from its default? If that flag couldn't be resolved, would the service still work?"

The same rule guards new changes and fixes existing code.

Feature flag default drift

STOP-GAP FOR NEW CHANGES

Agentic review

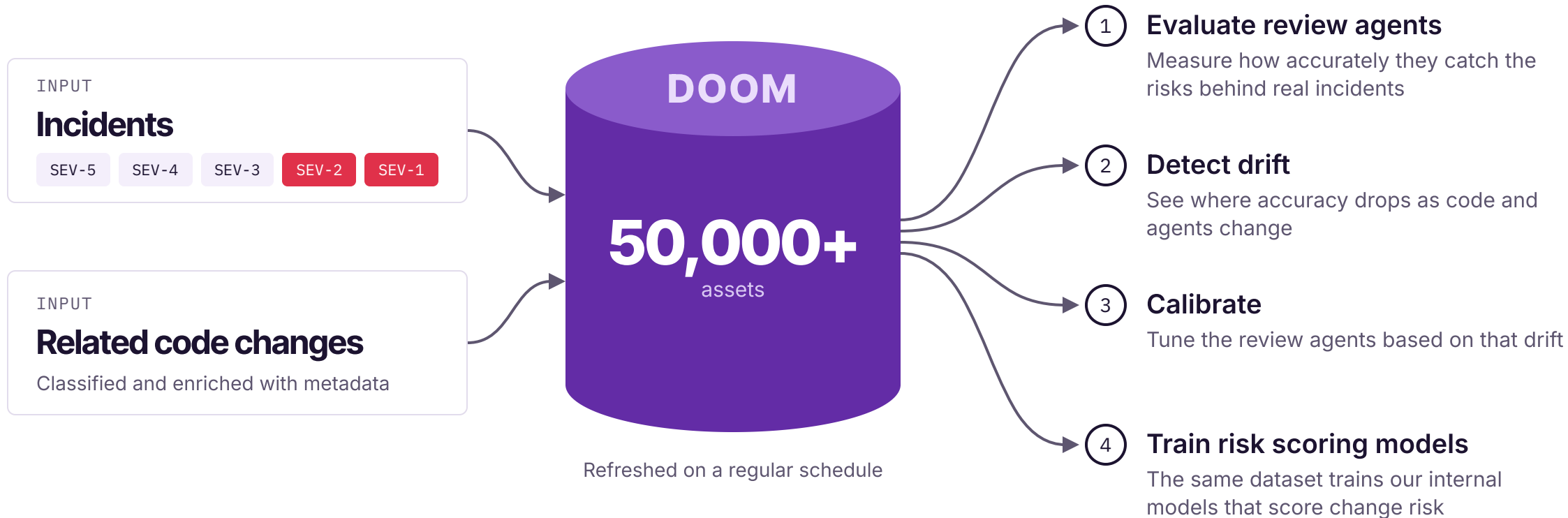
Catches any new change that reintroduces the risk. SRE teams put what they just learned from an incident straight into how changes are reviewed.

CAMPAIGNS ON EXISTING CODE

Agentic remediation

Finds existing instances across our services and rolls out fixes progressively. SRE teams use it to direct reliability improvement campaigns.

Every incident becomes evaluation data.



39% of incident-causing changes sit in the riskiest 15%.

DDRS

Datadog Diff Risk Score

TRAINED ON THE DOOM DATASET

53,000

production diffs

Teach an 8B model what Datadog code changes look like

39,239

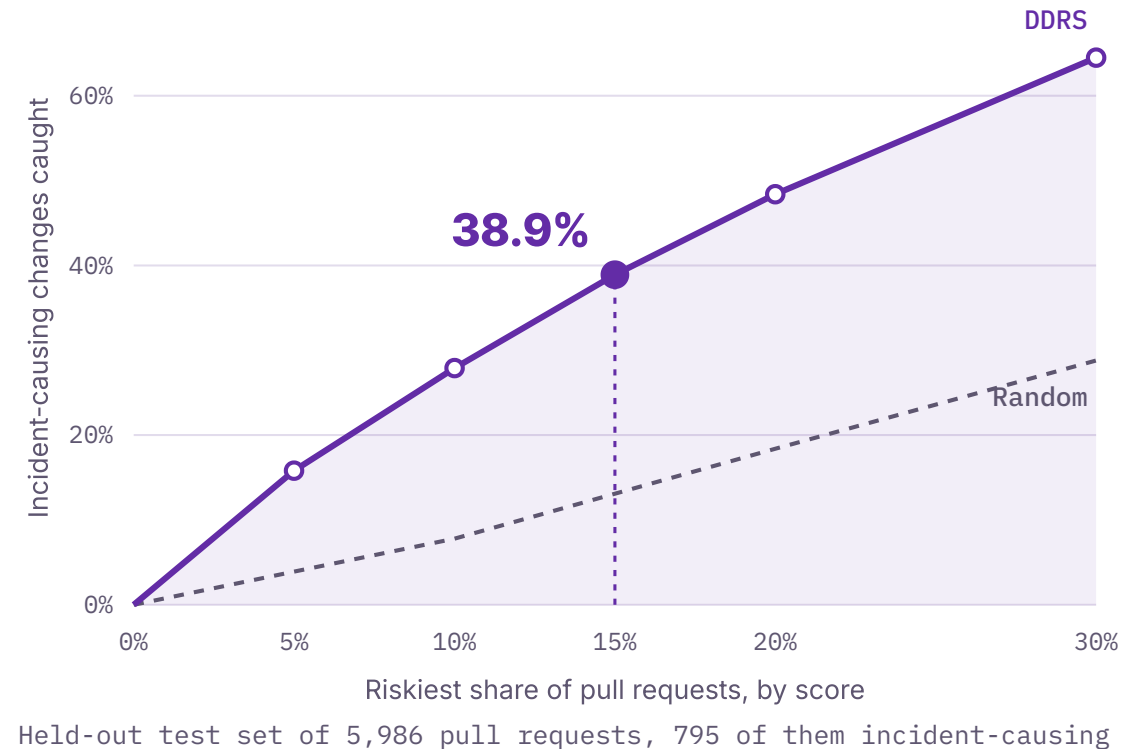
PR-to-incident links

Mined from 16,283 incidents in 175+ repositories

48,000

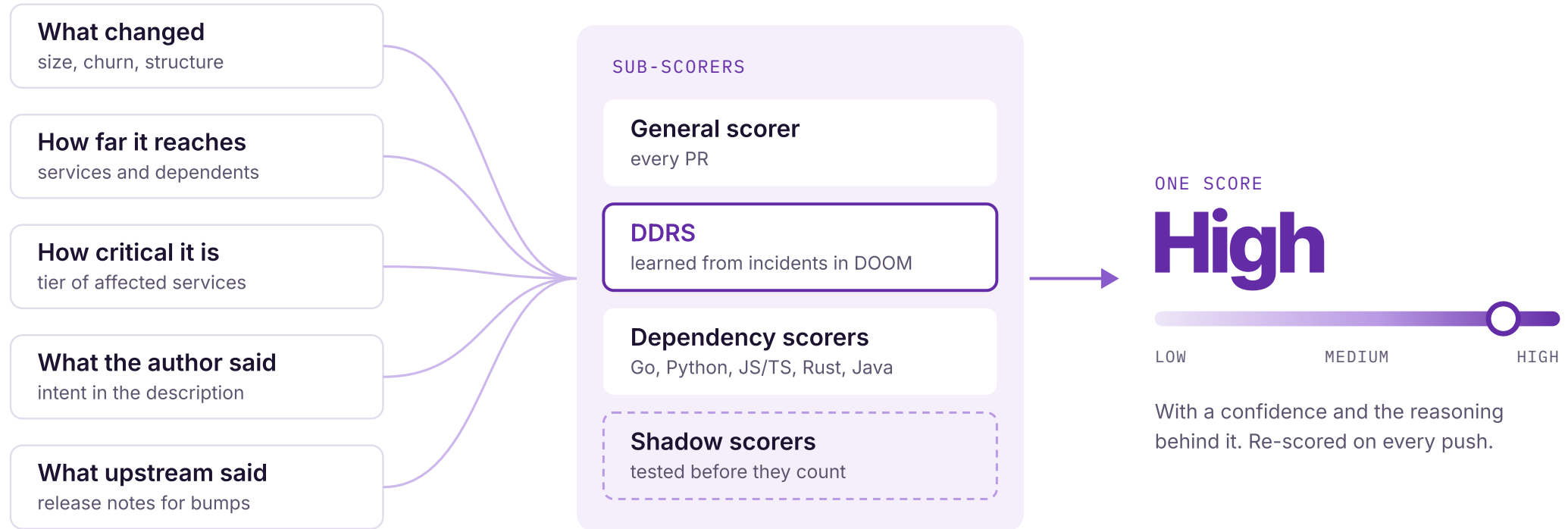
labeled examples

Fine-tune it to score each PR in about 2 seconds

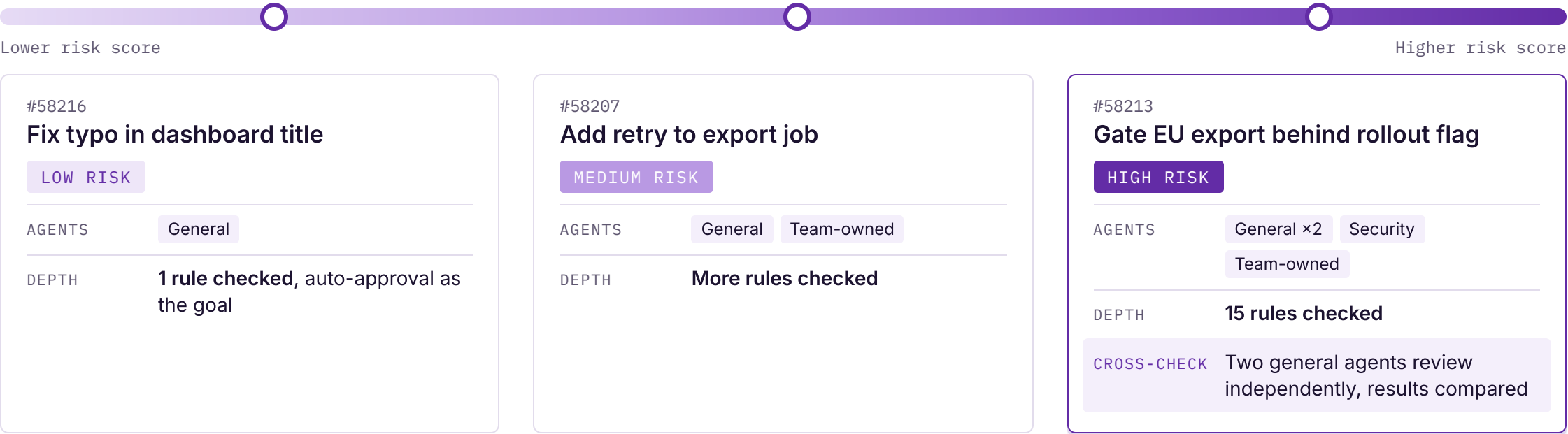


Every change gets one risk score, the moment it opens.

SIGNALS



The risk score decides how deep the agentic review goes.



From one outage to every change.

Learn from every outage

- 01 **Review each outage in depth**
Graded by severity, led by an incident commander, reviewed for quality
- 02 **Find the general risk**
Look beyond the one service that failed

Turn risks into rules

- 03 **Write it as a rule**
An invariant inside a reliability or threat pack
- 04 **Apply it everywhere**
Agentic review on new changes, remediation on existing code

Scale to every change

- 05 **Measure the agents**
The DOOM dataset evaluates review agents and trains DDRS
- 06 **Match review to risk**
The risk score sets how deep each review goes

Every outage makes the next change safer, and the next incident feeds the same loop.

CONCLUSION THANK YOU

Don't merge that. Here's why.

Reliability teams turned one outage into a rule. Agents now apply it to every change.

Thank you

Questions and discussion

Joris Bonnefoy · Staff Software Engineer, Datadog

Gate EU export behind rollout flag #58213

 Open joris-bonnefoy wants to merge 1 commit into `main` from `export/eu-gate`

Conversation 1 Commits 1 Checks 3 Files changed 1



agentic-review  requested changes

services/export/export.go

```
41 func Export(ctx context.Context, b Batch) error {  
42 +   if !flags.Bool(ctx, "eu-export.enabled", false) {  
43 +       return exportLegacy(ctx, b)  
44 +   }
```



agentic-review 

HIGH RISK

INV-3

reliability-pack / feature-flags

The default false doesn't match the live value, true in every datacenter. If the flag can't be resolved, every export silently falls back to the legacy path, the failure mode behind the resolver outage.

Suggested change

```
+   if !flags.Bool(ctx, "eu-export.enabled", true) {
```



Some checks were not successful

1 failing and 2 successful checks



risk-score / DDRS High risk



agentic-review / cross-check 15 rules, two general agents



agentic-review / reliability-pack 1 finding: INV-3