CIRCL
Computer Incident
Response Center
Luxembourg

GCVE.eu

The AI Paradox in Vulnerability Ecosystem

GCVE approaches

https://gcve.eu/ – https://vulnerability-lookup.org

Alexandre Dulaunoy – info@gcve.eu - @→ @gcve@social.circl.lu

GCVE Workshop Vulnopticon 2026

CIRCL https://www.circl.lu

The AI Paradox

The paradox

AI-assisted code review makes it easier to

discover vulnerabilities at scale.

The same increase in volume makes it harder to

analyse, coordinate and publish them at scale.

> The vulnerability ecosystem gets faster at producing findings than humans get faster at processing them.

What we see operationally

- At CIRCL, our role as both a **CNA** and **GCVE GNA** exposes us directly to the growing disclosure flow.
- As maintainers of open-source software such as **MISP**, we also receive a continuous stream of vulnerability reports, patches and security observations.
- AI-assisted code analysis lowers the cost of finding issues, including issues that previously required significant manual review.
- The bottleneck moves downstream: **validation, classification, advisory writing, coordination and publication**.

More findings are useful only if the vulnerability-handling ecosystem can absorb them.

The wrong response: remain passive

- More vulnerability findings do not automatically create more vulnerability intelligence.
- Manual-only processing does not scale with continuously increasing discovery throughput.
- Ignoring AI-assisted submissions does not remove them from the ecosystem.
- Blindly automating publication creates a different problem: **unreviewed and untraceable vulnerability data**.

> The objective is not autonomous vulnerability publication. The objective is to increase analyst leverage without losing accountability.

GCVE position: use AI, but make it explicit

- GCVE is designed around **attribution, provenance and autonomous publication**.
- AI-assisted processing should be treated in the same way: visible, attributable and reviewable.
- Automation should reduce repetitive analyst work while preserving the human decision points of CVD.
- Local and open tooling is important for pre-disclosure material and reproducible workflows.

> If AI contributes to vulnerability information, consumers should be able to know **where**, **how** and **with what level of human review**.

Three design principles

Assist

- Summarise
- Classify
- Draft
- Correlate

Trace

- Source evidence
- Model provenance
- Assumptions
- Review state

Keep control

- Local execution
- Open tooling
- Deterministic checks
- Human publication

GCVE treats AI as another producer of evidence and annotations, not as an authority.

Standardising AI Assistance

BCP-05-X-01: AI-Assisted Vulnerability Information Annotation

- Published extension of GCVE BCP-05.[1]
- Records when AI or automated processing contributed to **creation, enrichment, transformation or classification**.
- Supports annotations at **record level** or **field level**.
- Captures model information, GNA provenance, assistance level and human review status.

AI level	none	assisted	augmented / generated
Review	–	partial/full	none/partial/full

The consumer can distinguish human-authored data from AI-assisted or AI-generated data.

[1] https://gcve.eu/bcp/extension/gcve-bcp-05-x-01/

Why AI provenance matters

- AI output may be useful and still be incomplete or wrong.
- Different fields can have different provenance: description, severity, CWE, remediation, references, etc.
- A model name alone is insufficient: **who used it, what did it modify, and was the output reviewed?**
- BCP-05-X-01 gives downstream consumers machine-readable trust signals without pretending they guarantee correctness.

> Transparency is a prerequisite for using AI-generated vulnerability information operationally.

BCP-05-X-02: Patch-to-Vulnerability Generation Provenance

- Draft extension focused on transforming a **software patch into vulnerability metadata**.[2]
- Preserves the evidence chain: patch source, SHA-256, commit, subject and truncation state.
- Records generator/model, confidence, assumptions and classification rationales.
- Explicitly records whether the generated information is still a **draft requiring publication review**.

A patch is evidence, but a patch rarely contains every fact required for a publishable advisory.

[2] https://gcve.eu/bcp/extension/gcve-bcp-05-x-02/

Two complementary questions

BCP-05-X-01

- Where did AI contribute?
- Which model or system?
- What level of assistance?
- What level of review?

BCP-05-X-02

- What patch was analysed?
- What assumptions were made?
- Why this CVSS/CWE/CAPEC?
- Is the output still a draft?

One standard describes **AI involvement**; the other preserves the **evidence-to-advisory transformation**.

From BCPs to Running Code

Standards need implementations

- **gcve-eu-ai-extension**[3] demonstrates AI enrichment of CVE/GCVE records with BCP-05-X-01 provenance.
- **patch2vuln**[4] turns git-format patches into structured draft vulnerability advisories.
- Both are open source and designed around local processing.
- The goal is to make BCPs testable in real vulnerability-handling workflows, not only document them.

GCVE follows the principle: standardise what is already useful in running code, then make it interoperable.

[3] https://github.com/gcve-eu/gcve-eu-ai-extension
[4] https://github.com/gcve-eu/gcve-lab-patch2vuln

patch2vuln: analyst augmentation from the patch

Patch source, commit, metadata → **Local AI + deterministic processing** →

Title / description | CWE / CAPEC | CVSS v4 vector | Versions / fix / credits ⇓

Structured draft advisory + explicit provenance

This is the main GCVE tool currently used operationally at CIRCL to support vulnerability analysts.

One command: from a security patch to a draft advisory

```
python3 patch2vuln.py \
  --gcve-id GCVE-1-2026-20206 \
  --vendor misp \
  --product misp \
  https://github.com/MISP/misp/commit/382188d2f.patch
```

- Reads a patch from a file, URL or standard input.
- Uses one or more locally hosted Ollama models for structured analysis.
- Validates model output with Pydantic.
- Calculates the numeric CVSS v4.0 score deterministically from the proposed vector.
- Preserves analysis provenance and marks the result as a draft for review.

Reference: https://gcve.eu/2026/09/18/introducing-patch2vuln-v1.0-streamlining-vulnerability-advisory-creation-from-git-patches/

patch2vuln is deliberately not an autonomous publisher

- The model proposes technical interpretation; the analyst remains responsible for publication.
- Affected versions, deployment assumptions and real-world prerequisites may not be visible in the patch.
- CWE/CAPEC and CVSS vectors can be ambiguous and require review.
- Patch text itself is untrusted input and may contain misleading or prompt-injection-like content.
- Assumptions and rationales are retained specifically to focus analyst review.

Automate the repetitive work; keep the security judgement and publication decision with the analyst.

Format agnostic output

- The analysis should not be trapped in one vulnerability format.
- `patch2vuln` can generate a CVE Record Format 5.2 draft and complementary GCVE, CSAF 2.1 and OSV records.
- GCVE provenance travels with the advisory as extensions where appropriate.
- This matches the broader GCVE strategy: **identifier and provenance models should coexist with existing advisory formats**.

Patch $\rightarrow$ **Analysis** $\rightarrow$ { `CVE`, `GCVE`, `CSAF`, `OSV` }

The smaller building block: AI record enrichment

- `gcve-eu-ai-extension` fetches an existing CVE/GCVE record and enriches it using a configurable local Ollama model.
- It can generate an analyst-oriented summary, recommendation, confidence and caveats.
- The enrichment is accompanied by BCP-05-X-01 annotations describing AI involvement and review state.
- The same repository also demonstrates enrichment with VLAI severity classification.

AI-generated content and AI provenance should be produced together.

Efficient Models for Classification

Not every AI task needs a large language model

- Vulnerability handling includes many repetitive classification tasks.
- For those tasks, small specialised models can be faster, cheaper and easier to reproduce.
- CIRCL developed **VLAI**, an NLP-based severity classification approach trained from Vulnerability-Lookup data.[5]
- The severity classifier predicts **Low / Medium / High / Critical** from a textual vulnerability description when an official score is not yet available.

> Use general models where reasoning over heterogeneous evidence is useful; use specialised models where classification can be learned efficiently.

[5] https://www.vulnerability-lookup.org/user-manual/ai/

VLAI: control the complete model pipeline

Vulnerability-Lookup feeds → AI-ready datasets → local training → local inference → Vulnerability-Lookup API

- Datasets are built from multiple vulnerability sources and updated regularly.
- Models and training tooling are open and reproducible.
- The English severity model is based on a fine-tuned RoBERTa classifier.
- Inference is exposed through a local model-serving layer and integrated into Vulnerability-Lookup.

The important capability is not only the model: it is owning the dataset → training → serving → operational-use chain.

VLAI and GCVE provenance fit together

- A VLAI severity classification can enrich a vulnerability before an official score exists.
- The enrichment can be stored as an additional assertion rather than overwriting authoritative source data.
- BCP-05-X-01 can identify the VLAI model as the source of the classification and state the review level.
- Consumers remain free to use, ignore or compare the assertion with later vendor/CNA scoring.

Federation makes AI-derived vulnerability intelligence additive: new analysis does not need to erase existing evidence.

A Scalable CVD Model

A practical human + automation pipeline

Discovery → Evidence → AI-assisted analysis → Structured draft

⇓

Human validation → Coordination → Publication → Updates / sightings / enrichment

- Automation is strongest before and around analyst review.
- Provenance survives the complete processing chain.
- Publication authority remains with the CNA/GNA/vendor/CSIRT operating the CVD process.

What changes for the vulnerability analyst?

Manual-heavy workflow

- Read the patch
- Draft description
- Find CWE/CAPEC
- Build CVSS vector
- Extract credits
- Build JSON records

AI-assisted workflow

- Review evidence
- Challenge assumptions
- Verify scope/versions
- Correct classifications
- Decide coordination
- Approve publication

The analyst spends less time formatting vulnerability information and more time validating the parts that require security expertise.

The paradox becomes an opportunity

- AI increases vulnerability discovery throughput.
- That creates pressure on CNA/GNA/PSIRT/CSIRT workflows.
- The answer is not to slow discovery; it is to improve the processing pipeline.
- GCVE adds open standards for **AI provenance** and **evidence provenance**.
- Open-source tools such as `patch2vuln` and VLAI make those standards operational.

The goal is a vulnerability ecosystem where automation increases human capacity instead of reducing trust.

Takeaways

1. **The vulnerability firehose is real**: AI-assisted discovery moves the bottleneck toward analysis and coordination.
2. **AI assistance must be attributable**: BCP-05-X-01 exposes how AI contributed and how it was reviewed.
3. **Generated advisories need evidence provenance**: BCP-05-X-02 keeps patch, assumptions, rationales and draft state together.
4. **Running code matters**: patch2vuln supports analysts end-to-end; VLAI supports efficient classification at scale.
5. **Humans remain authoritative**: automation prepares, enriches and prioritises; analysts validate and publish.

Thank you for your attention

- GCVE BCPs: https://gcve.eu/bcp/
- patch2vuln: https://github.com/gcve-eu/gcve-lab-patch2vuln
- AI extension: https://github.com/gcve-eu/gcve-eu-ai-extension
- VLAI and datasets: https://www.vulnerability-lookup.org/user-manual/ai/
- Questions: info@gcve.eu
- GCVE **@@gcve@social.circl.lu**