enisa
EUROPEAN UNION AGENCY FOR CYBERSECURITY

CERT-EU

Joint Advisory on the Use of Frontier AI Models for Code Review

An experiment conducted by the European Union Agency for Cybersecurity, with the support of the Cybersecurity Service for the Union Institutions, Bodies, Offices and Agencies, in the framework of the EU action plan on cybersecurity and artificial intelligence.

NOVEMBER 2026

CONTACT

For contacting the authors, please use ai@enisa.europa.eu.

For media enquiries about this paper, please use press@enisa.europa.eu.

AUTHORS

ENISA, CERT-EU

LEGAL NOTICE

This publication represents the views and interpretations of ENISA, unless stated otherwise. It does not endorse a regulatory obligation of ENISA or of ENISA bodies pursuant to Regulation (EU) No 2019/881.

ENISA has the right to alter, update or remove the publication or any of its contents. It is intended for information purposes only and it must be accessible free of charge. All references to it or its use as a whole or in part must show ENISA as its source.

Third-party sources are quoted as appropriate. ENISA is not responsible or liable for the content of the external sources including external websites referenced in this publication. Neither ENISA nor any person acting on its behalf is responsible for the use that might be made of the information contained in this publication. ENISA reserves its intellectual property rights in relation to this publication.

COPYRIGHT NOTICE

© European Union Agency for Cybersecurity (ENISA), 2026

This publication is licenced under CC-BY 4.0. 'Unless otherwise noted, the reuse of this document is authorised under the Creative Commons Attribution 4.0 International (CC BY 4.0) licence (https://creativecommons.org/licenses/by/4.0/). This means that reuse is allowed, provided that appropriate credit is given and any changes are indicated'.

AI tools were used in a limited capacity to support editorial refinement. All outputs were reviewed and validated by subject-matter experts. No AI-generated content was used without substantive human oversight.

English PDF Web TP-01-26-026-EN-N ISBN: 978-92-9204-819-8 DOI: 10.2824/7940736





CERT-EU

Joint Advisory on the Use of Frontier AI Models for Code Review

Table of contents

Executive summary 5

Background 6

Code audit review using frontier artificial intelligence 7

- The Vigilo harness 8
- European Union Agency for Cybersecurity skills 8
- Our approach 8
- Observations and takeaways 9

Lessons learned 13

Annex 14

Executive summary

ENISA, the European Union Agency for Cybersecurity, in the context of key action 4 of the EU action plan on cybersecurity and AI ([1]), conducted an experiment to assess how frontier artificial intelligence (AI) models could support security reviews of open-source software maintained by Union institutions, bodies, offices and agencies (Union entities).

ENISA conducted this experiment with the help of CERT-EU, the Cybersecurity Service for the Union Institutions, Bodies, Offices and Agencies, which provided Vigilo, a specialised security harness it developed. The experiment also gave ENISA the opportunity to test its own AI skills in code auditing.

In total, 27 codebases (25 already open-source and 2 soon to become open-source) were analysed. The reviews examined source code only: the models were given no information about production deployments, and no live applications were tested.

Across the 27 codebases, the models that were used identified 220 potential issues, which were shared with the respective code owners.

Assessing AI-generated findings requires time and expert effort, and this work is still ongoing. At the time of publication, code owners had completed their assessment of 88 findings, confirming 95 % of them as true positives, and four new common vulnerabilities and exposures (CVE) identifiers (CVE IDs) had been assigned. To conclude, for the limited subset of AI-identified issues on which code owners gave feedback, ENISA found a high true-positive rate.

The experiment shows that AI can help accelerate code review but cannot replace human judgement. Without deployment context, models may overestimate the practical impact or severity of an issue, while the volume of findings they produce can place a significant review burden on code owners. The tools also involved trade-offs: general-purpose coding agents with security skills or plug-ins were generally faster and less costly, while the specialised harness performed additional analysis steps and reduced the number of unsubstantiated findings, at the cost of more time and more tokens.

Organisations considering similar AI-assisted code review tasks should work in a secure, isolated testing environment, ensure that qualified human experts review the results and their severity, and follow established coordinated vulnerability disclosure (CVD).

1 Key Action 4 – ENISA in cooperation with relevant Union entities will issue guidance, recommendations, advisories and best practices on the protection against AI-powered threats and for the secure integration of AI in cybersecurity operations (as of Q3 2026)' (https://digital-strategy.ec.europa.eu/en/library/eu-action-plan-cybersecurity-and-artificial-intelligence).

Background

As part of key action 4 of the EU action plan on cybersecurity and AI [2], ENISA, with support from CERT-EU, used frontier AI models to analyse open-source software projects maintained by Union entities.

The main objective was to experiment with frontier AI models for security-focused code auditing. The experiment also provided an opportunity to test Vigilo, an AI harness developed by CERT-EU [3], and two ENISA-developed skills: Code Auditor, for code auditing, and Findings Browser Report, for reporting findings. The aim was to understand whether such customised tools could improve the performance of frontier AI models when reviewing source code.

For this purpose, ENISA has been granted access, through structured access programs, to various frontier AI models with advanced cybersecurity capabilities, including OpenAI GPT-5.5 Cyber, GPT-5.6 Cyber, GPT-6 Astra and Anthropic Mythos 5. ENISA used these models to scan the source code of the selected open-source software maintained by Union entities. No live applications were tested.

Drawing on suggestions from CERT-EU, ENISA compiled a non-exhaustive list of open-source software maintained by Union entities, focusing on applications deployed in Union entities' infrastructure or underpinning essential services for Union entities and citizens. The selection was agreed with the respective code owners.

Moreover, a selected group of Union entities was invited to submit for review the code of critical, internet-facing open-source applications that they maintain.

The experiment gave participating Union entities an opportunity to identify and address weaknesses in their codes before adversaries could find and exploit them with AI-powered tools.

2 'Key Action 4 – ENISA in cooperation with relevant Union entities will issue guidance, recommendations, advisories and best practices on the protection against AI-powered threats and for the secure integration of AI in cybersecurity operations (as of Q3 2026)' (https://digital-strategy.ec.europa.eu/en/library/eu-action-plan-cybersecurity-and-artificial-intelligence).

3 CERT-EU, Vigilo GitHub repository, accessed 27 October 2026, https://github.com/certeu/vigilo.

Code audit review using frontier artificial intelligence

Under key action 4 of the EU action plan, ENISA set out to explore whether AI can be used safely, repeatedly and responsibly in real-world code review, and whether doing so improves the security of publicly exposed services.

Since June 2026, ENISA has been running AI-assisted source code reviews of pre-selected open-source software developed, used and maintained by Union entities and the broader European digital ecosystem.

The reviews used the models, harnesses and skills listed in Tables 1 and 2, in various combinations.

Table 1. Models used

MODEL	ACCESS PROGRAMME	REFERENCE
OpenAI GPT-5.5 family (including cyber) OpenAI GPT-5.6 family (including cyber) OpenAI Astra model	Daybreak	https://openai.com/daybreak/
Anthropic Mythos 5	Project Glasswing	https://www.anthropic.com/glasswing

Table 2. Harnesses used

HARNESS	TYPE	DEVELOPED BY	REFERENCE
OpenAI's Codex client	Client	OpenAI	https://github.com/openai/codex
Codex Security plug-in	Plug-in	OpenAI	https://openai.com/business/plugins/codex-security/
OpenCode agentic client	Client	Anomaly	https://github.com/anomalyco/opencode/
Vigilo	Client	CERT-EU	https://github.com/certeu/vigilo
ENISA Findings Browser	Skill	ENISA	https://github.com/enisaeu/agentic-skills/tree/main/skills/findings-browser-report
Code Auditor	Skill	ENISA	https://github.com/enisaeu/agentic-skills/tree/main/skills/code-auditor

The Vigilo harness

Vigilo is an AI-powered penetration-testing pipeline developed in-house by CERT-EU and released as open-source in September 2026. Instead of relying on a single AI agent, it orchestrates several specialised agents that share their discoveries through a common knowledge base. Each agent focuses on one vulnerability class, such as injection, authentication, authorisation, cryptography or supply-chain risks in dependencies.

Vigilo can run in two modes: white box and grey box. In white-box mode, it analyses the source code and, optionally, a running instance of the application. In grey-box mode, it tests only the running application, acting as an authenticated external attacker.

In both modes, an adversarial critique step re-examines every finding, challenging its severity and merging duplicates, so that the final report contains only findings the critique step considers substantiated. Where source code is available, Vigilo can also generate a patch for each such finding automatically, on a separate branch for the code maintainers to review.

Organisations can deploy Vigilo on their own infrastructure, either as a command-line tool or as a multi-user web platform, using cloud-based or on-premises AI models. Testing a running application should only be carried out on systems that the organisation owns or is explicitly authorised to test. As with any AI-assisted assessment, code owners should validate findings and proposed patches before applying any fix.

In this experiment, Vigilo was used in white-box mode on source code only, confirming findings at the source level rather than against live applications.

European Union Agency for Cybersecurity skills

The following agentic skills were developed by ENISA, utilising AI-powered tools, following industry standards to allow compatibility between multiple agentic clients.

Code Auditor

Code Auditor instructs an AI agent how to perform a security-focused source code audit. In this experiment, it guided the agent through a structured, multiphase audit and had it record the evidence supporting each finding as it went. The skill also includes reporting templates, both for summary reports and for individual findings, so that results can easily be communicated to the agent's operators and shared with others, such as code owners.

Findings Browser Report

Findings Browser Report provides instructions, scripts and templates for turning existing code audit results, such as findings and evidence, into a single, self-contained, static HTML report that is easy to browse. In this experiment, it was used to produce readable, interactive reports from the raw output of the various code audit tools. The agent reads the audit outputs and extracts the relevant information into a structured JavaScript Object Notation file, which a Python script then converts into the HTML report. Because this task is relatively simple, it can be performed by a smaller, lower-cost model.

Our approach

1. Environment set-up

First, a secure, isolated testing environment was set up so that the AI models could not access any systems, data or services beyond the scope of the experiment (Figure 1). The AI harnesses and supporting tools ran in a sandbox, separated from the host environment [4], and the abovementioned plug-ins, skills and reporting templates were added to it.

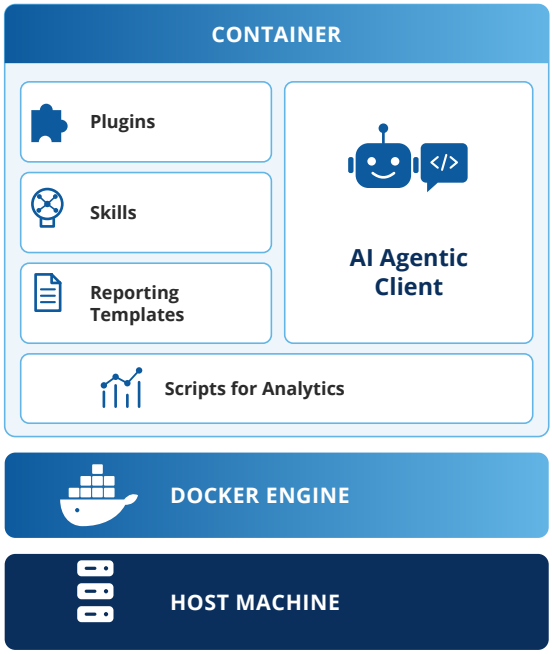


Figure 1. AI local lab environment.

4 Organisations setting up a similar environment should also consider the additional security measures described in 'Observations and takeaways' below.

2. Codebase selection

Codebases were selected to cover applications deployed in Union entities' infrastructure or underpinning essential services for Union entities and citizens. Open-source applications were prioritised because their code is publicly available and can therefore also be scanned for weaknesses by malicious actors.

3. Code audit

The automated AI-powered code audits were run using the models and tools mentioned above, using the source code only, with no information about production deployments (e.g. configuration or infrastructure). No live applications were tested.

4. Review and validation of selected findings

Rather than treating the model output as confirmed findings, for a limited number of codebases ENISA reviewed the AI-identified findings, developed controlled proofs of concept and manually tested the findings to assess their impact and reproducibility.

Furthermore, where applicable, ENISA deduplicated the output and notified the code owners, following their CVD policy, including all the relevant information that led to CVE ID assignment.

5. Reporting of findings

Before reporting, ENISA removed duplicate findings. All findings were then reported in line with CVD principles. For codebases maintained by Union entities, disclosure was coordinated through CERT-EU [5]. Through CERT-EU's trusted network of contact points, the findings were promptly shared with the respective code maintainers. Maintainers were provided with all the information needed to assess each finding and asked to confirm whether it was a true positive and to validate its severity.

6. Feedback collection and analysis

ENISA collected and analysed the feedback that code owners had provided by the time of publication. For each finding, it compared the AI's assessment with the code owners' assessment of whether the finding was a true positive and how severe it was. The results are presented in 'Observations and takeaways' below, and Table 3 summarises the aggregated, anonymised statistics.

Table 3. Aggregated statistics

CODEBASES	AI-IDENTIFIED FINDINGS	ACKNOWLEDGED BY CODE OWNERS	CVE ID ASSIGNMENTS	TOTAL TOKENS CONSUMED (M)
27	220 for expert triage	88	4	611,486,042

As the source code reviews were conducted in August 2026 and used several frontier AI models [6], the actual cost varies. It is roughly estimated that the total financial cost of all the scans conducted was around EUR 6 648.90 and that the average code audit cost per 10 000 lines of code was about EUR 132.27. These figures are indicative only; ENISA was granted access to the frontier AI models free of charge under the providers' access programmes.

Observations and takeaways

1.

A properly secured testing environment is an integral part of AI-assisted code auditing. Modern AI security tools can read repositories, create files, execute commands, install dependencies and call remote services, so the environment must limit what they can reach. Security measures should include network segmentation, clearly defined trust boundaries, access control and monitoring. Organisations should also plan for the effort involved: setting up a secure environment takes considerable time, and the environment then needs continuous monitoring.

5 CERT-EU acts on behalf of Union entities as the equivalent of the CVD coordinator designated under Article 12(1) of Directive (EU) 2022/2555 (the NIS 2 Directive), pursuant to Article 13(3), point (g), of Regulation (EU, Euratom) 2023/2841 of the European Parliament and of the Council of 13 December 2023 laying down measures for a high common level of cybersecurity at the institutions, bodies, offices and agencies of the Union.

6 https://developers.openai.com/api/docs/models/gpt-5.5; https://developers.openai.com/api/docs/models/gpt-5.6-luna; https://developers.openai.com/api/docs/models/gpt-5.6-terra; https://developers.openai.com/api/docs/models/gpt-5.6-sol; https://developers.openai.com/api/docs/models/gpt-6-astra; https://developers.openai.com/api/docs/models/gpt-6-sol; https://developers.openai.com/api/docs/models/gpt-6-luna; https://platform.claude.com/docs/en/models/mythos-5/overview; https://platform.claude.com/docs/en/models/mythos-5/overview.

2.

The severity of a finding can rarely be determined from source code alone. To assess it, a model needs to know what is reachable and with which privileges, which components are trusted, what external controls exist and which risks have been accepted. Without information about how the software behaves in production, an AI agent may miss critical context and misjudge severity.

3.

Figure 2 shows how the 220 AI-identified findings were distributed across severity levels (critical, high, medium and low), as rated by the AI tools.

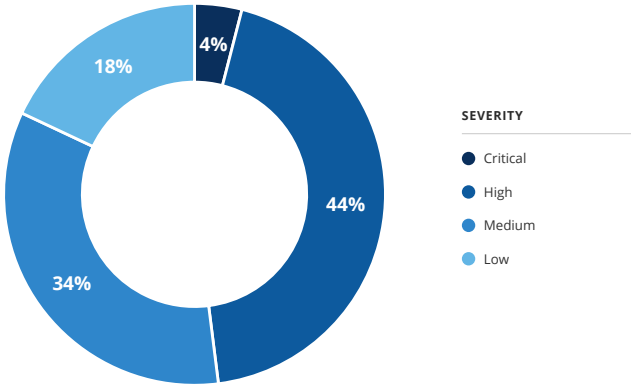


Figure 2. AI-identified findings by severity level.

4.

All AI-identified findings were shared with the code owners in August 2026. By the time of publication, code owners had completed their assessments of 40 % of them (88 findings), as Figure 3 shows. The remaining findings were still being assessed. Assessing AI-generated findings, particularly those not pre-validated, requires dedicated expert time from code owners.

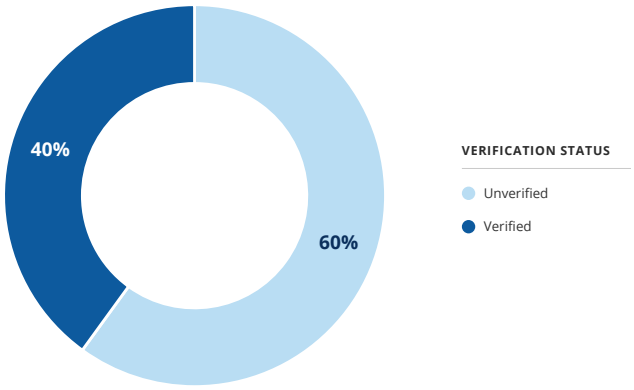


Figure 3. Acknowledged findings by code owners.

5.

Of the 88 findings assessed by code owners, 95 % were confirmed as true positives (Figure 4). The remaining 5 % were classed as false positives or not applicable, mostly because the AI tools lacked information about the runtime environment, which was deliberately withheld from them.

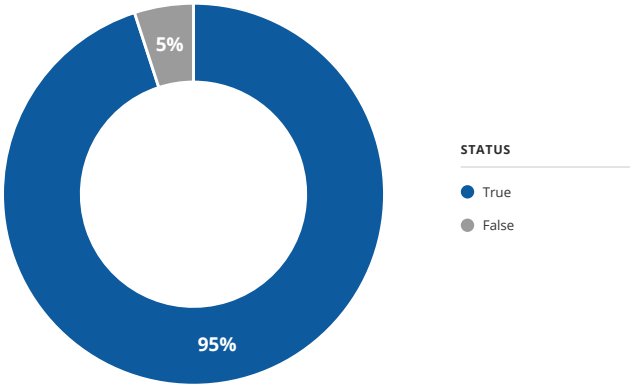


Figure 4. True positive versus false positive for the acknowledged AI findings.

6.

The AI-identified findings, after having been reviewed by the code maintainers, were eventually rated as 26 % high severity, 34 % medium, 29 % low and 11 % informational (Figure 5).

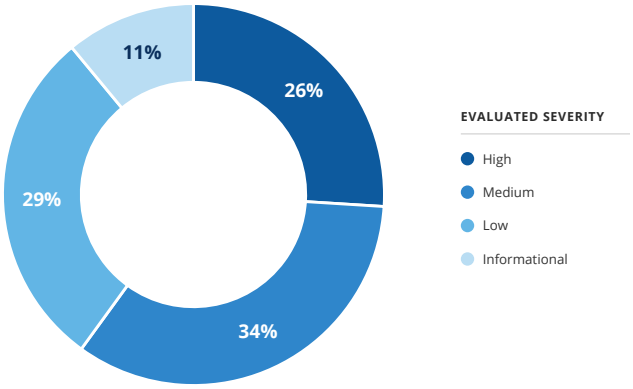


Figure 5. Severity levels of the acknowledged AI findings.

7.

Figure 6 depicts the new severity levels of the AI-identified findings after their acknowledgement by the code owners.

Based on the scanned codebases, Figure 6 depicts a slight tendency of the AI models to overrate the severity level of their findings. The difference in the severity levels before and after the human review of the AI findings stresses the need for human validation of the AI-assisted code audit outputs.

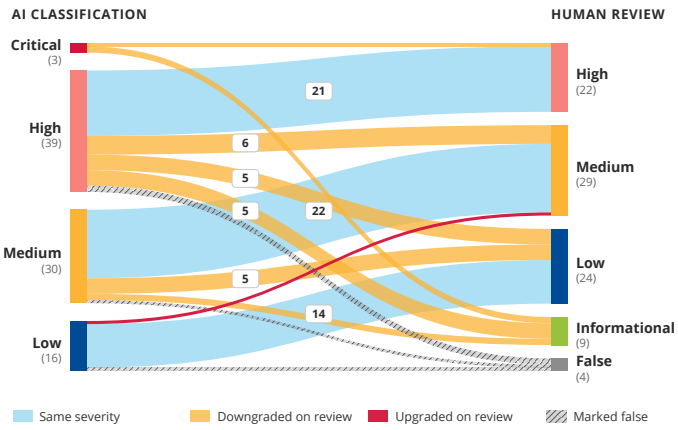


Figure 6. AI findings severity before and after human review.

8.

Figure 7 compares, for each type of AI tool used, how code owners re-rated the severity of confirmed findings. Code owners kept the severity assigned by Vigilo, the specialised security harness developed by CERT-EU, in 74 % of cases, compared with 60 % for general-purpose coding agents with a security skill or plug-in.

This suggests that the additional steps of a specialised harness, such as validating findings with an adversarial agent, produce severity ratings closer to the code owners' own assessment.

However, in both cases, most changes were downgrades, which confirms that the severity ratings from AI tools need human review before they are relied upon.

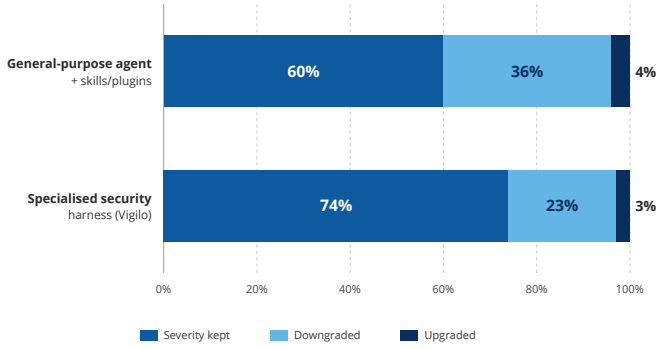


Figure 7. Severity change after code owner review, by type of tool.

9.

Feedback from code owners was essential to assess both the accuracy of the AI-identified findings and the reliability of their severity ratings. ENISA and CERT-EU thank the code owners who reviewed findings and shared their assessments. Their feedback also led to four new CVE IDs being assigned to one of the scanned codebases as a result of the experiment.

10.

As Figure 8 shows, general-purpose coding agents with a security skill or plug-in were usually faster and cheaper than the specialised security harness. Vigilo took longer and cost more, as expected; it runs additional steps, such as validating findings with an adversarial agent and generating patches, which some skills and plug-ins do not perform by default. Runtime and cost grow with the size of the repository, but size is not the only factor: the type and purpose of the codebase (e.g. a library or package versus a full web application), its architecture and the number of entry points to analyse also affect how long an audit takes and how much it costs.

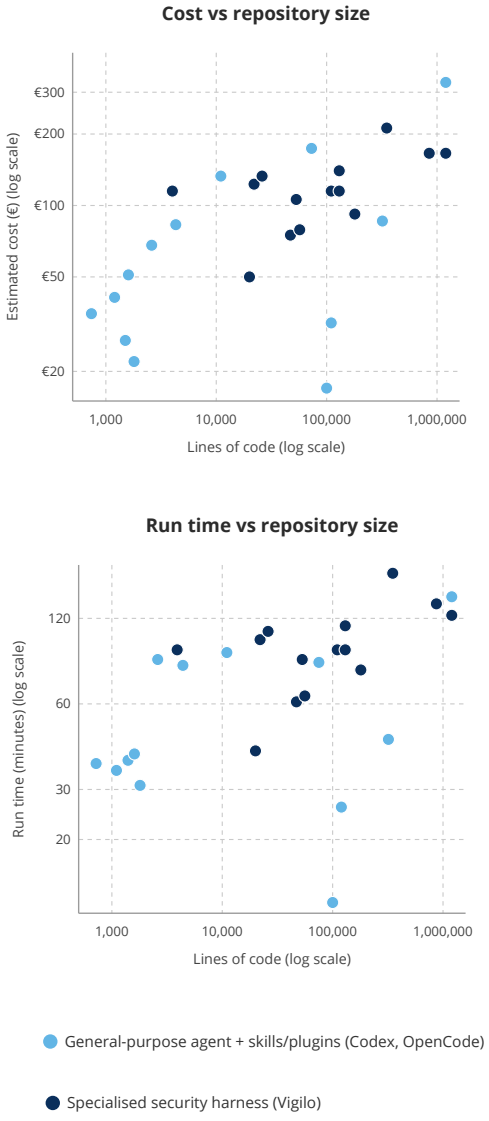


Figure 8. Cost and runtime of the AI-assisted code audits by repository size.

11.

Frontier AI models can accelerate vulnerability research. They can also help analysts generate proof-of-concept exploits in a controlled environment, to establish whether a suspected weakness can be reproduced before it is disclosed.

12.

Findings should always be reported through the appropriate CVD channel: for Union entities, through CERT-EU as their CVD coordinator; for community-maintained open-source projects, in line with each project's own CVD policy. Reports that bypass these channels risk being disregarded and may expose users if a vulnerability becomes public before a fix is available.

13.

The experiment highlights the critical role of humans in the loop on both sides of the disclosure process. Researchers should deduplicate and, as far as possible, verify AI output before reporting it, so that code owners are not overwhelmed by duplicates or false positives. Where full verification requires deployment context that only the code owner has, this should be made clear when the findings are shared. Code owners, in turn, need sufficient time and resources to assess AI-identified findings with due diligence.

Lessons learned

The experiment's observations are consistent with the trend CERT-EU described in April 2026: AI is changing the economics of vulnerability discovery, making weaknesses faster and cheaper to find, while defenders must still triage and correctly fix every finding ([7]).

The experiment showed that AI can find weaknesses and vulnerabilities in software in a very short time. Validating, prioritising and fixing them, however, still requires time and expert effort and cannot currently be accelerated to the same extent.

Frontier AI models proved effective at finding real vulnerabilities: most of the findings assessed by code owners were confirmed as true positives. However, AI models have a limited understanding of the environment in which the software runs, which led them to overestimate the severity of some findings. Therefore, AI-identified vulnerabilities need to be validated by people with the right expertise and context, such as penetration testers and the software's developers, who can confirm whether each finding applies and how severe it is.

Frontier AI models can produce large numbers of findings quickly and speed up code review but human experts remain essential, and accountable, for removing duplicates, setting priorities, confirming that findings can be reproduced, deciding on severity and disclosure, and ensuring that AI tools are operated safely. The aim should be to reduce repetitive work and improve efficiency while preserving expert judgement.

The experiment also showed that different types of AI tools suit different purposes. General-purpose coding agents with a security skill or plug-in were faster and cheaper, while the specialised security harness took longer but produced severity ratings closer to the code owner's own assessment. The two approaches therefore complement each other: a specialised security harness is better suited to an in-depth audit of a code repository, while a general-purpose coding agent with a security skill or plug-in can provide a useful baseline for routine changes, for example when a pull request is made.

This is increasingly relevant as AI coding assistants are used more widely to write software: the volume of new code to review is growing, and AI-generated code can itself introduce vulnerabilities. Although this experiment did not examine whether the reviewed code was AI generated, routine AI-assisted security checks at the point where code is introduced may offer a practical way to keep pace.

Finally, human review takes time, and skilled experts are scarce, but such a review remains essential: a fully automated patching cycle could introduce new defects or fail to fix the underlying issue. If vulnerabilities continue to be identified at this pace, the shortage of experts able to verify AI-identified findings will become an increasingly serious constraint. Future efforts should therefore focus on relieving code owners, for example by developing ways to triage AI-identified findings and consolidate overlapping results before they reach code owners for verification.

Finally, the experience showed the need for organisations to focus on education to upskill their employees, so that cybersecurity experts can obtain the maximum benefits from AI.

7 'AI is changing the economics of vulnerability discovery. Defenders should adapt now', CERT-EU website, 21 April 2026, https://www.cert.europa.eu/blog/ai-vulnerability-discovery-defenders-must-adapt.

Annex

Drawing on the experience gained in this experiment, the following seven-step process can be used to integrate AI across the vulnerability management life cycle.

1. **Select or develop agentic skills and testing harnesses** tailored to the target environment, and run them in a secure, isolated environment.
2. **Use reputable AI models with cybersecurity capabilities** to discover and classify vulnerabilities across the application or codebase, either through a specialised security harness or through a general-purpose coding agent with a security skill or plug-in.
3. **Deduplicate, triage and prioritise findings** based on severity, relevance and potential impact, with qualified human experts validating their severity.
4. **Use AI-assisted proof-of-concept exploits,** in a controlled environment, to validate findings and demonstrate real-world risk.
5. **Report confirmed issues through the appropriate CVD channel:** for Union entities, through CERT-EU as their CVD coordinator; for other projects, in line with their own CVD policy.
6. **Generate remediation guidance and patches** for validated vulnerabilities, for code owners to review before applying them.
7. **Monitor token usage and operational costs** to evaluate efficiency and scalability.

The cycle should be repeated continuously, using results and feedback to improve subsequent testing, and evaluating the relevance of newer models as they become available.

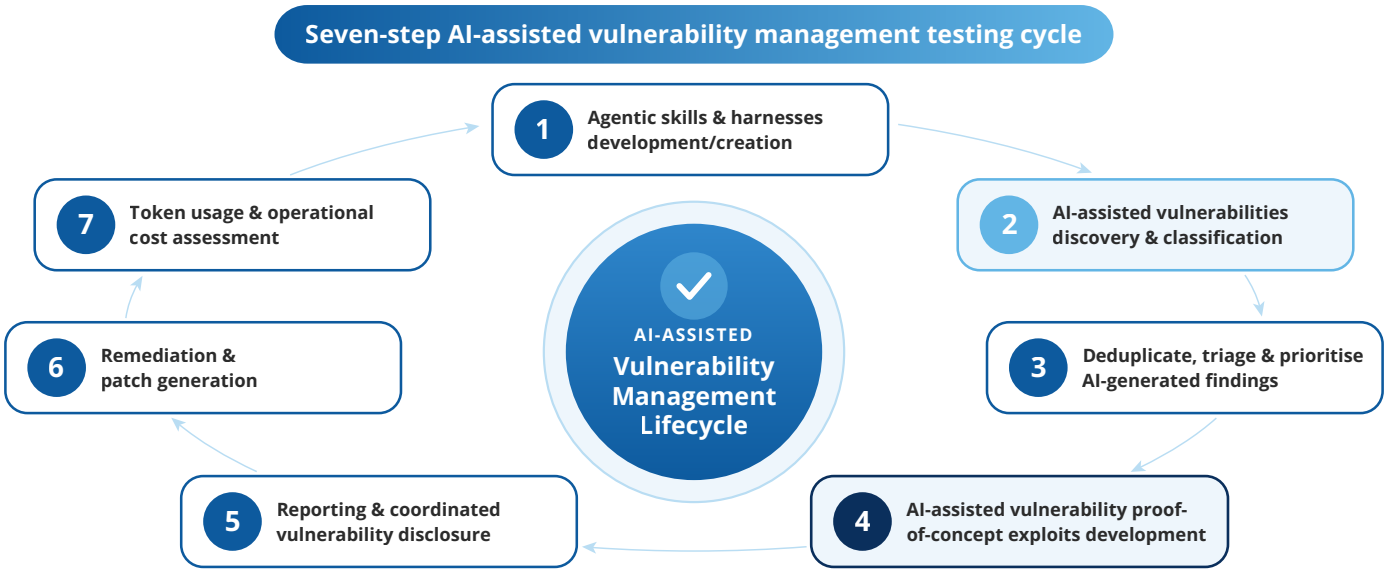


Figure 9. AI-assisted vulnerability management life cycle.

ABOUT ENISA

The European Union Agency for Cybersecurity, ENISA, is the Union's agency dedicated to achieving a high common level of cybersecurity across Europe. Established in 2004 and strengthened by the EU Cybersecurity Act, the European Union Agency for Cybersecurity contributes to EU cyber policy, enhances the trustworthiness of ICT products, services and processes with cybersecurity certification schemes, cooperates with Member States and EU bodies, and helps Europe prepare for the cyber challenges of tomorrow. Through knowledge sharing, capacity building and awareness raising, the Agency works together with its key stakeholders to strengthen trust in the connected economy, to boost resilience of the Union's infrastructure, and, ultimately, to keep Europe's society and citizens digitally secure. More information about ENISA and its work can be found here: www.enisa.europa.eu.

ABOUT CERT-EU

CERT-EU is the Cybersecurity Service for the European Union institutions, bodies, offices and agencies (Union entities). Its legal basis is Regulation (EU, Euratom) 2023/2841 laying down measures for a high common level of cybersecurity at the institutions, bodies, offices and agencies of the Union (OJ L, 2023/2842, 18.12.2023) (the 'Cybersecurity Regulation'). CERT-EU was established in 2011. It serves as an inter-institutional provider, governed by the Interinstitutional Cybersecurity Board, serving all the Union institutions, bodies, offices and agencies (Union entities).

TP-01-26-026-EN-N

ISBN 978-92-9204-819-8

ENISA
European Union Agency for Cybersecurity

Athens Office
Agamemnonos 14
Chalandri 15231, Attiki, Greece

Brussels Office
Rue de la Loi 107
1049 Brussels, Belgium

enisa.europa.eu

Publications Office
of the European Union