



ENISA Technical Advisory on AI-assisted software development

Fields marked with * are mandatory.

ENISA Technical Advisory on AI-assisted software development

Thank you for taking part in this open consultation. We are seeking feedback on our [draft Technical Advisory on AI-assisted software development](#). Your input will help ensure that the content is practical, relevant, and reflective of real-world practices.

This survey invites you to comment on key aspects of the advisory, including its overall approach, coverage of risks and threats, and the clarity and usefulness of its recommendations. We are also interested in whether the advisory is sufficiently concise and where content could be clarified, refined, expanded or removed.

Your feedback, whether based on hands-on experience, security expertise or general familiarity with AI-assisted software development, is highly valued. The results of this consultation will help inform the final revision of the Technical Advisory.

The consultation will remain open until the **15th of October 2026**.

Thank you in advance for your valuable contributions.

Data Protection

Authors may voluntarily provide their name, surname, email, and affiliation as part of this survey. If the author gives explicit consent, their name, surname, and affiliation may be included in the acknowledgements of the

final report; however, inclusion is not guaranteed.

Data collected within this survey will be deleted 3 months after the processing of the submissions.

☒ I have read and understood the [Data Protection Notice](#).

Respondent info

Name and surname (optional)

Javier Valiño

Current job title (optional)

Program Manager

Affiliation (optional)

Open Regulatory Compliance

☒ If names are included in the acknowledgements, I am happy for my name and affiliation to be included. I understand that inclusion is not guaranteed.

Please provide any relevant context about your work, organisation, or experience that may help us better understand your feedback.

The Open Regulatory Compliance (ORC) Working Group is a global, industry-led collaboration helping organisations understand and implement open source compliance strategies under evolving regulations like the Cyber Resilience Act (CRA) without compromising open source innovation. We unite manufacturers, open source stewards, and maintainers to create practical, community-driven best practices for secure and resilient software.

Feedback

1) Overall, how useful do you find the draft Technical Advisory on AI-assisted software development?

7

Please explain your rating (optional)

Note: If you suggest a change, please indicate the relevant section or subsection, quote or describe the current text, and provide an example of the proposed revised or additional text.

The four actions (identify, specify, verify, record) are clear and fit well with the CRA's lifecycle-based approach. However, the advisory assumes an organisation that can impose requirements on all parts of the development process. For example, Section 4.2 describes a skill "with its use required through organisational processes". Open source projects usually cannot control contributors' development environments; their effective control begins when a change is submitted.

Suggested change:

Relevant Section: 1. Scope and context.

Current Text: "Where AI-assisted software development is used in these activities, the same secure by design approach should be considered in how AI-assisted tasks are defined, guided, reviewed and verified."

Proposed Additional Text (insert after the Current Text):

"For open source projects, this advisory distinguishes between the contribution phase, where maintainers can recommend practices but usually have no control over contributors' development environments, and the acceptance phase, which begins when a change is submitted (for example as a pull request) and in which the project can apply the review, security and quality checks it controls. Because maintainers often cannot tell whether a contribution was produced with AI assistance, t

2) How accurately and clearly does Section 2 represent a typical AI-assisted software development workflow and its main components?

7

Please explain your rating (optional)

Note: If you suggest a change, please indicate the relevant section or subsection, quote or describe the current text, and provide an example of the proposed revised or additional text.

Section 2 captures the basic components well, but two aspects need attention.

First, Figure 1 and Section 2.2.2 assume that whoever directs the assistant or agent also reviews and accepts its output. In open source projects these are often different people, and the point of submission is a trust boundary.

Second, the section underestimates how agent autonomy expands the trust boundary. Section 2.2.2 notes that more autonomous uses may require additional controls "beyond the scope of this advisory". Where a control can be implemented once, that is preferable to each project recreating it. The agent provider should provide defaults for the capabilities it adds. Repository and CI/CD platforms and agent providers should each provide secure defaults for the capabilities they control, rather than each project recreating them.

Suggested change 1:

Relevant Section: 2.1, Figure 1.

Current Text: Figure 1 shows a single "Developer" who both defines the task and performs "review, accept, reject".

Proposed Revised or Additional Text:

Figure 1: for open source projects, show the "review, accept, reject" step as performed by a maintainer or other reviewer when the change is submitted (for example as a pull request), with a trust boundary between the contributor's environment and the project.

Insert below Figure 1, after the agent-led bullet in Section 2.2.2:

"In open source projects, the person directing the assistant or agent and the person who accepts the resulting change may be different. The reviewer may not have access to the prompts or context used to produce the change, so acceptance should depend on the checks applied at the trust boundary, not on reconstructing the development process."

Suggested change 2:

Relevant Section: 2.1.2 Coding assistants and agents.

Current Text: "Depending on their permissions and configuration, they may inspect repository content, propose or apply changes, run commands and tests, install or update dependencies, generate documentation, or prepare commits and pull requests."

Proposed Additional Text (insert after the Current Text):

"Where agents can run commands, hold credentials or modify repository, build or deployment configuration, those agents and their tools and integrations sit inside the software supply chain. Where controls can be implemented effectively once at the platform or tooling layer, secure defaults there are preferable to requiring each project to recreate them. Repository and CI/CD platforms and agent providers should provide proportionate protections for the capabilities they control, such as sandboxed execution, scoped and short-lived credentials, and restrictions on access to secrets and write permissions when processing untrusted content."

3) How well do the risks, failure modes and adversarial threats described in Section 3 reflect realistic scenarios in AI-assisted software development?

7

Please explain your rating (optional)

Note: If you suggest a change, please indicate the relevant section or subsection, quote or describe the current text, and provide an example of the proposed revised or additional text.

Section 3 identifies the main categories of risk well, including tampering with project instructions and skills, uncontrolled autonomy, and prompt injection through repository content. We suggest four additions. First, Table 2 covers AI-generated changes that are hard to review, but not changes that are larger than the task requires.

Second, in a public repository, a change to instruction files such as AGENTS.md or to skills can affect the agents of many contributors and downstream users, and agents themselves need to be treated as security-relevant components.

Third, repository history alone is not reliable evidence of process, because it can be rewritten or generated after the fact.

Fourth, Table 3 does not cover attackers flooding a project with AI-generated pull requests, issues or vulnerability reports.

Suggested change 1:

Relevant Section: 3.1, Table 2.

Current Text: Table 2 (Non-adversarial risks and failure modes), which ends with the row "Weak tests, review or evidence".

Proposed Additional Text: Add a new row at the end of Table 2.

Risk / failure mode: "Unnecessary or excessive code"

Example: "The assistant or agent produces more code than the task requires, for example by duplicating existing functionality, adding unnecessary abstractions, or reimplementing what the platform, an existing dependency or an approved library already provides. This increases the review and maintenance burden and can enlarge the attack surface."

Suggested change 2:

Relevant Section: 3.2 Adversarial threats.

Current Text: "If they are malicious, outdated or tampered with, they can influence how an assistant or agent generates/modifies code, selects dependencies, invokes tools or explains security claims."

Proposed Additional Text (insert after the Current Text):

"In public repositories, instruction files and skills may be read by the agents of many contributors and downstream users, so an unnoticed change to them can have a much wider reach than a change to ordinary code. Assistants and agents are themselves security-relevant components: they can fail accidentally or be manipulated through prompts, context, tools, dependencies or credentials."

Suggested change 3:

Relevant Section: 3.3, Table 4.

Current Text: Table 4 (Governance and assurance risks), which ends with the row "Unsupported security claims".

Proposed Additional Text: Add a new row at the end of Table 4.

Governance / assurance risk: "Unverifiable evidence"

Example: "Records exist, but a reader cannot tell whether the agent under review produced them, or whether they describe the artefact that was delivered. For example, repository history can be rewritten or generated after the fact."

Suggested change 4:

Relevant Section: 3.2, Table 3, Denial of service.

Current Text: "An attacker manipulates inputs or context so that an agent performs disruptive or resource-intensive actions, e.g. repeatedly triggering builds or corrupting build configuration."

Proposed Additional Text (insert after the Current Text):

"An attacker may also flood a project with AI-generated pull requests, issues or vulnerability reports, overwhelming the maintainers' review capacity which could be better used."

4) How well do the good practices proposed in Section 4 address the relevant risks, failure modes and threats.

3

Please explain your rating (optional)

Note: If you suggest a change, please indicate the relevant section or subsection, quote or describe the current text, and provide an example of the proposed revised or additional text.

Section 4 rightly builds on established practice: once a contribution reaches an open source project, the recommendations largely come down to the review, testing and automated checks that projects already use for submitted changes. Four changes would make the practices work better for open source projects. First, open source projects can recommend skills and instruction files but usually cannot enforce their usage on all contributors, and Section 4 does not say how to protect these files in public repositories. Second, review capacity is only half of the problem; the other half is volume. AI should help developers write less code, reuse more and keep changes small, not produce changes we would never have accepted from a human developer. Third, human review does not scale linearly with AI-generated volume, so scarce human review effort should be focussed where it adds most value. Fourth, Section 4.4 could be read as requiring evidence of whether AI was used, which maintainers usually cannot know, and it does not say who writes the record. A record the agent under review could edit is not independent evidence of its actions, and signing alone does not make it independent.

Suggested change 1:

Relevant Section: 4.2.

Current Text: "Annex A0 provides a practical example on how security requirements can be encoded in reusable skills."

Proposed Additional Text (insert after the Current Text):

"Where contributors are outside the organisation's control, as in many open source projects, skills and instruction files can be recommended but their use cannot be enforced. Security requirements should therefore be applied when contributions are accepted (Section 4.3). Repository platforms should provide practical and proportionate mechanisms that projects can use to protect repository-level instruction files and skills, for example protected paths or required review."

Suggested change 2:

Relevant Section: 4.2.

Current Text: "This helps apply the same security expectation consistently across projects, tools and development teams." (the last sentence of the example box)

Proposed Additional Text: Insert at the end of Section 4.2, after the example box:

"Expectations should also cover the necessity and size of changes. AI-assisted changes should meet the same expectations as other contributions: reuse existing code, platform functionality and dependencies where appropriate, avoid unnecessary duplication and abstractions, and keep changes small and focussed."

Suggested change 3:

Relevant Section: 4.3.

Current Text: "These requirements should be proportionate to the risk of the activity, considering ... the level of autonomy granted to it."

Proposed Additional Text (insert after the Current Text):

"Human review does not scale linearly with the volume of AI-generated changes, and approval without meaningful review can create false assurance. Review should therefore be risk-based: automate routine checks where practical, and focus scarce human review effort on architecture, policy, exceptions, risk acceptance and high-impact components such as cryptography, authentication, parsers and build configuration."

Suggested change 4:

Relevant Section: 4.4.

Current Text: "Where appropriate, organisations can move towards more structured evidence generation, ... integrated into CI/CD pipelines."

Proposed Additional Text (insert after the Current Text):

"The evidence described in this section concerns how a change was verified and accepted, and applies whether or not AI assistance was used to produce it. Where repository or CI/CD platforms perform relevant checks or approvals, they should make the resulting records available automatically so that projects can reuse them as evidence rather than recreate them manually."

5) How useful is the illustrative example in Annex A in demonstrating how the proposed approach can be applied in practice?

7

Please explain your rating (optional)

Note: If you suggest a change, please indicate the relevant section or subsection, quote or describe the current text, and provide an example of the proposed revised or additional text.

Annex A is a clear and practical example, and ENISA's publication of its skills in a public repository is welcome; we encourage ENISA to invite contributions to it from open source communities. We suggest two changes, both about evidence.

First, the example skill's expected output in A.4 lists what to record, but does not require the expected output to identify the supporting evidence a reviewer can check. A later reader should be able to check the supporting evidence rather than take the record on trust.

Second, a skill cannot make its own evidence independent: a record written by the agent under review is not independent evidence of its actions, and signing alone does not make it independent. Independent records can also support the due diligence required of manufacturers integrating third-party components under CRA Art. 13 (5).

Suggested change 1:

Relevant Section: A.4, the "Expected output" section of the example skill.

Current Text: "checks performed and results;"

Proposed Revised Text (to replace the Current Text):

"checks performed and results, with references to supporting evidence where available, such as CI/CD records;"

Apply the same change to the evidence template referenced in the skill.

Suggested change 2:

Relevant Section: A.5 Limitations of skills.

Current Text: "They do not replace secure engineering judgement, human review, vulnerability management or accountability for accepted changes."

Proposed Additional Text (insert after the Current Text):

"A skill cannot by itself make the evidence it requests independent. Where independent evidence is needed, it should preferably be generated automatically by the system that performs the relevant check or approval, outside the control of the assistant or agent. This places evidence generation where it can be done most reliably and efficiently, without adding manual work for maintainers or creating barriers to contributions, including security fixes."

6) Any additional comments or suggestions? (optional)

Note: If you suggest a change, please indicate the relevant section or subsection, quote or describe the current text, and provide an example of the proposed revised or additional text.

We would like to make two further points.

First, the advisory should say clearly who its practices are for. Continuous automated verification consumes CI /CD resources and human review consumes the scarce resource of maintainers' time. The advisory should not present manufacturer-oriented practices as expectations for open source software stewards, who have specific tailor-made obligations as set out in Art. 24. Under Recital 21, manufacturers that integrate free and open source components could also initiate or finance the voluntary security attestations foreseen in Art. 25.

(Question 1 addresses the attack surface of an open source project; this point here concerns the separate regulatory distinction between open source software stewards and manufacturers.)

Second, the advisory does not address long-term maintainability. AI-assisted changes may not preserve the rationale behind design decisions, making later maintenance and security fixes harder. In one open source project in our group, a record of design decisions kept beside the code and updated in the same commit has proved valuable for long-term security maintenance.

Suggested change 1:

Relevant Section: 1. Scope and context.

Current Text: "It does not represent a prescriptive implementation framework and its application should be adapted to the product's intended use, context and associated risks and to the applicable requirements."

Proposed Additional Text (insert after the Current Text):

"The practices in this advisory are primarily addressed to manufacturers and other organisations that control their development processes. These recommended practices should be distinguished from the obligations of open source software stewards under Art. 24 of the CRA. Where manufacturers integrate third-party open source components, they remain responsible for exercising the due diligence required by Art. 13(5); the voluntary security attestation programmes foreseen in Art. 25 are intended to facilitate that due diligence."

Suggested change 2:

Relevant Section: 4.4 Record the supporting evidence.

Current Text: "Where appropriate, evidence should be generated and retained to show what was changed, why it was changed, which checks were performed, what the results were, and who reviewed or approved of the changes."

Proposed Additional Text (insert after the Current Text):

"Evidence should also be useful to future maintainers, not only for assurance or audit purposes. Where significant design decisions or security rationale are already documented, the evidence should reference and preserve that information rather than require duplicate records."

If you would like to share any supporting material (e.g., examples, guidelines, research, or other feedback), please upload it here (optional)

Thank you for providing your feedback and for your support in our endeavors.

Please feel free to get in touch by sending us an email at: product_security@enisa.europa.eu

Contact

[Contact Form](#)