Co-funded by the European Union

ECCC EUROPEAN CYBERSECURITY COMPETENCE CENTRE

OCCTET

Open Source Compliance Comprehensive tools and resources designed to simplify and streamline the CRA compliance process for SME, allowing them to tackle the complexities of OSS compliance.

Project Title: Open Source Compliance Comprehensive tools and resources designed to simplify and streamline the CRA compliance process for SME, allowing them to tackle the complexities of OSS compliance.
Project Acronym: OCCTET
Grant Agreement / Contract No.: 101190474
Program: DIGITAL Europe Programme; DIGITAL-ECCC-2024-DEPLOY-CYBER-06
Instrument: DIGITAL JU SME Support Action
Granting Authority: European Cybersecurity Industrial, Technology and Research Competence Centre
Project Start Date: 1 November 2024
Project Duration: 24 months

Deliverable Number: D3.2
Deliverable Title: BasicChain-Del
Deliverable Type (DOA): DEM - Demonstrator, pilot, prototype
Deliverable Type (content): Beta release of all the tools in a downloadable and installable form
Work Package: WP3 – Build Open Source Tools to Automate Evaluation
Task Number(s): -
Dissemination Level: PU – Public
Due Date (DoA): 31 October 2025
Actual Submission Date: 31 October 2025
Version: 1.1 (PR1 Revision)

Lead Beneficiary: BS- Bitsea
Main Author(s): Hans-Juergen Schumacher - BS
Contributing Partner(s): BS - AboutCode - DO - ECL
Reviewer: ECL - Eclipse Foundation Europe

Legal Notice

This deliverable has been produced within the OCCTET project (Grant Agreement No. 101190474) funded under the Digital Europe Programme.

Document History

Version	Date	Issued By	Status	Comments
1.0	07-10-2025	BS	Final	Approved deliverable for submission to the European Commission.
1.1	18-02-2026	ECL	Final	Addressing the recommendations from the Review Report

Executive Summary

Executive Summary

The **D3.2–BasicChain-v1.1 (BasicChain-Del)** report announces the beta release of the core open-source toolchain for the **OCCTET** project, fulfilling **Milestone M7**. This initiative addresses the critical need for compliance tools among Small and Medium-sized Enterprises (SMEs) following the EU's **Cyber Resilience Act (CRA)**, which mandates "security by design," vulnerability management, and documentation (like SBOM and VEX).

The OCCTET project delivers a holistic, integrated, open-source solution comprising:

- **ORT Toolchain:** Provides automated dependency analysis, license scanning, vulnerability identification, policy evaluation, and generation of standard SBOMs (SPDX, CycloneDX), accessible via a central server and GUI.
- **Occtet-Curator:** An AI-supported web application for human-in-the-loop audit workflows, inventory management, and planned VEX generation to streamline compliance.
- **Federated Database (VulnerableCode, etc.):** Delivers essential, aggregated security vulnerability and reference data to enrich the SBOMs.

This beta release confirms the core components are available and can be used sequentially, demonstrating the fundamental workflow for automated CRA compliance readiness. Future efforts (WP4) will focus on testing, deepening interoperability, refining vulnerability workflows, and enhancing the User Experience (UX/UI) to sustainably lower the compliance burden for European SMEs.

Keywords: Cyber Resilience Act (CRA), SMEs, FOSS, Open Source Compliance, Software Bill of Materials (SBOM), OSS Review Toolkit (ORT), Occtet-Curator, Vulnerability Exploitability eXchange (VEX), Federated Database, VulnerableCode, Dependency Analysis

Table of contents

1 Introduction to D3.2 OCCTET project deliverables

This section describes the requirements for the deliverables in the OCCTET project and their fulfilment at different stages of the project.

1.1 Grant agreement

As per the grant agreement, the objectives of this Work Package 3 are as follow:

- Develop an open source toolkit to streamline the generation of SBOMs, facilitating the efficient identification and cataloguing of software components within digital assets.
- Implement automated evaluation mechanisms to swiftly assess software vulnerabilities, enabling prompt detection and prioritisation based on their severity and potential impact.
- Enable effective remediation strategies by offering actionable insights and recommendations to address software vulnerabilities through patch management, version updates, and configuration adjustments.
- Assess the security posture of open source components, evaluating their security practices, exposure to known vulnerabilities, and overall maintenance status to ensure the use of secure components in software development.
- Automate compliance reporting with the generation of critical documents such as SBOMs and Vulnerability Exploitability Exchange (VEX) documents, aiding in adherence to Critical Risk Assessment (CRA) standards and simplifying regulatory compliance processes.

As per the Grant agreement, the task 3.1 in the whole OCCET project is as follows:

T3.1 Discovery: Create automatic open source dependency and code analysis solution to generate accurate SBOMs.

The first and essential step to compliance is to understand which third-party and open source software packages are used in a software product or device. This inventory is also known as an SBOM.

In this task, we build a solution with open source tools to discover and create accurate software package inventories as well as ingest SBOMs from third-party tools in SPDX and CycloneDX formats. The core discovery includes static and dynamic dependency analysis and will be supported by key FOSS projects such as ORT and ScanCode. The outcome is the automated creation of different types of SBOMs for multiple programming tech stacks. The discovery and dependency analysis will be implemented to detect the technologies used in a codebase such that limited code changes are necessary to perform the analysis. Furthermore, the solution will be packaged such that limited user efforts are needed to perform such an analysis. As part of this discovery solution, we will provide tools to efficiently review and curate the results of the analysis for accuracy and will implement a new AI/ML-assisted review of scan results and SBOM for correctness. This and other utilities will be created to ensure the discovery of high quality and actionable SBOMs which are essential to support a seamless and automated process in the compliance pipeline.

As per the Grant agreement, the task 3.3 in the whole OCCET project is as follows:

T3.3 Open source software supply chain: Develop tools to automate triage, evaluation of security posture and remediation of vulnerabilities.

Based on the Reference Metadata, and the discovered SBOM, the next step is to produce an enriched SBOM with known vulnerable software packages, and to assess if a vulnerability applies and the risks associated with the vulnerability. In particular, it is essential to determine whether a patch or fixed version of the vulnerable code is available, or if the vulnerability is exploitable and reachable in the context of how the software is used. Automating this evaluation empowers SMEs to integrate FOSS components into their products with greater confidence, faster, efficiently, and securely, lowering the compliance and due diligence burden. It is also essential that Open Source Stewards (such as Eclipse or Apache Foundations) have the proper tools to enable their critical support to foster compliance at large in their respective FOSS project ecosystem for the direct benefit of SMEs in particular that will be able to reuse a Steward's analysis.

This task focuses on creating advanced, automated tools to streamline this assessment process for the compliance of open source software components. By leveraging these tools, we aim to significantly reduce the manual triage effort of these evaluations, while also enhancing their accuracy. The development of these tools will align closely with the Conformity Assessment Specifications outlined in W2. This is particularly crucial concerning the compliance requirements of the Cyber Resilience Act (Article 17b), ensuring that FOSS components do not compromise the cybersecurity posture of SMEs' products.

A first step supported by this task is to enrich SBOM packages inventory so that they can be automatically mapped to security vulnerability information. This is done using Package URLs (or PURL) and the Reference Metadata for open vulnerabilities and package origin scan and data delivered in this Work Package. This includes reusing shared scans and SBOM information stored in the Reference Data to accelerate the process of review and assessment and avoid redoing work or analysis already shared by the user community.

A second step supported by this task is to manage the effective triage of mapped security vulnerability, based on multiple factors and the determination of mitigation or patching course of action to remediate the issue. Some key factors are derived from the sharing of mitigation and remediation solutions by other users of the solution that contributed enriched metadata to the Reference Data.

Finally, this task also includes mapping of the inventory to other secondary software supply chain risks such as sustainability, maturity and quality of a software component using data sourced from the Reference data, and the computation of appropriate risk scores.

As to the milestone deliverable on 31 October 2025 (Grant Agreement, Annex A, list of deliverables, list of milestones) requirements:

- D3.2 BasicChain-Del WP3
- Demonstrator, pilot, prototype
- Dissemination level: public
- Milestone M7: BasicChain, deliverable at the end of month 12
- Description: Beta release of all the tools in a downloadable and installable form
- Milestone requirement: All the tools from all tasks can be called sequentially to demonstrate basic integration

1.2 Milestone (M6) 30.4.2025

The Deliverables for **D3.1 FedDB-Beta-Del** WP3 were issued on **30 April 2025**, as per the published deliverable report. The key elements of that previously completed milestone are the initial **FederatedCode Demonstrator** focused on software package vulnerabilities.

1.3 Milestone (M7) 31.10.2025

The key tools chosen for the task are:

- OSS Review Toolkit (ORT)
- ORT Server

Related tools

- OCCTET curator

The current working versions of the tools are available, licensed under open source licensed at the following repositories:

- https://github.com/oss-review-toolkit/ort
- https://github.com/eclipse-apoapsis/ort-server
- https://github.com/Bitsea/Occtet-Curator

For a detailed report, see the following sections.

1.4 Remaining milestones after 31.10.2025

As per the Grant agreement, the task 3.4 in the whole OCCET project is as follows:

T3.4 Report: Create open source documentation generator and CRA compliance reporting tool.

To comply with the CRA, the results of the risks and vulnerabilities assessment must be shared with constituents including upstream affected FOSS projects, downstream users and customers, and government and regulatory agencies. In this task, we build a comprehensive open source reporting solution to generate outboundSBOMs, VEX (Vulnerability Exploitability eXchange), and attestations to support effective communication with these constituents and achieve the corresponding proper CRA compliance from handling software vulnerabilities.

As per the Grant agreement, the remaining deliverables under Work Package 3:

Deliverable D3.3 Toolkit-Demo-Del WP3

Primary partner: Bitsea

Description: Demonstrator, pilot, prototype

Dissemination level: Public

At the end of month 17 (31.3.2026)

Deliverable D3.4 FedDB-Final-Del WP3

Primary partner: ABCD

Description: Demonstrator, pilot, prototype

Dissemination level: Public

At the end of month 17 (31.3.2026)

The final version of the toolkit is to be delivered under Work Package 4, as D4.1, by 31.10.2026.

1.5 OCCTET outlook for end of the project, 31.10.2026

The toochain solution has progressed as planned, and concurrent improvements alongside the project have further supported the project, extending and benefiting the results of the project. The decision to use open source licensed tooling has clearly been proven, as concurrent improvements done by others than the project itself, benefit also the project results, all being part of the same open source projects. There is a great outlook for the usage of the tooling solution also after the project, considering the interest and usage not only by project participants but also non-project parties.

The project is moving now to the phase of testing the tooling, mainly happening under Work Package 4, and incrementally improving the tooling based on the results of the testing. We are seeing signals of both organic adoption from iconic open source projects as well as adoption from SMEs due to OCCTET project outreach. OCCTET project outreach has helped us understand potential areas for improvement, and that work will continue as part of the testing part of the project.

The fact that the OCCTET tooling is open source, and developed as an open source project, has ensured and will continue to ensure the long-term utility and availability of the results of the project.

2 Background

In today's interconnected digital economy, Free and Open Source Software (FOSS) forms the invisible backbone of nearly every product with digital elements. From routers and medical devices to mobile apps and enterprise platforms, developers across all sectors routinely rely on FOSS to accelerate development, reduce costs, and foster innovation. It is estimated that between 90% to 99% of modern software products incorporate open source components making FOSS not only ubiquitous but mission-critical to Europe's digital infrastructure.

However, this widespread dependence on FOSS also exposes systemic vulnerabilities. Security incidents such as the Log4Shell vulnerability and the recent XZ Utils backdoor have highlighted how a single flaw in a widely used open source library can cascade through thousands of products across sectors. More recently, high-profile software supply chain attacks such as the s1ngularity malware embedded in the Nx npm package, or the NPM supply chain attack on popular packages like chalk and debug have exposed how a single compromised component can cascade across thousands of downstream applications. Moreover, many companies, especially Small and Medium-sized Enterprises (SMEs) are unaware of the full extent of their open source dependencies, let alone how to manage their compliance and cybersecurity risks.

Amid growing cyber threats, the EU must increasingly treat open source base technologies as digital infrastructure. Securing open source software means securing the foundation of all our digital systems. As noted in the EU Sovereign Tech Fund Feasibility Study, the scale of this need is vast because open source code is integral to nearly every digital asset. Its open nature enables innovation, but it also requires deliberate security and maintenance strategies. Like water management systems or transport networks, open source software must be maintained with public investment because failures can have cascading consequences.

FOSS plays a role comparable to roads and bridges in physical infrastructure, capital markets in economic systems, and water in environmental systems. In each of these analogies, open source code forms a critical enabler — and its maintenance offers high-return investment. According to the Linux Foundation, 70–90% of modern software stacks are composed of open source code. Replacing this would cost the European economy an estimated €8.8 trillion — underscoring FOSS not just as a technical asset but a pillar of Europe's economic independence, innovation, and competitiveness.

Recognizing these challenges, the European Union has introduced landmark legislation such as the Cyber Resilience Act (CRA) and the NIS2 Directive, which impose strict obligations on manufacturers to ensure that digital products including the third-party components they incorporate — meet essential cybersecurity and documentation requirements. Notably, Article 10(4) of the CRA states that manufacturers must ensure all integrated software, including FOSS, does not compromise the product's cybersecurity. The Cyber Resilience Act (CRA) represents a landmark shift in the European Union's approach to cybersecurity. By introducing mandatory cybersecurity and vulnerability management obligations for manufacturers of digital products and software, the CRA establishes a horizontal legal framework that directly affects the development and distribution of nearly all products with digital elements (PDEs) in the EU internal market. This includes not only hardware devices but also general-purpose and embedded software components.

The CRA is grounded in the principle that cybersecurity must be embedded "by design" throughout the product lifecycle—from development to deployment and post-market

surveillance. This introduces new obligations such as vulnerability reporting, technical documentation, secure update mechanisms, and clear communication with end-users. In particular, Annex I and Article 10 of the CRA emphasize transparency, secure software development practices, and detailed technical documentation of software components and known vulnerabilities

While these legal developments are crucial for improving digital trust, they pose practical difficulties for SMEs. Without dedicated legal or compliance teams, many SMEs struggle to verify software origins, fulfill license obligations, or track vulnerabilities across their dependency trees. Manual compliance processes are time-consuming and costly, creating a compliance gap that threatens to leave smaller actors behind.

The OCCTET Project (Open source Compliance Comprehensive Tools and Efforts for Trustworthiness) was created to directly address this gap. Funded by the European Commission through the Digital Europe Programme, OCCTET is a strategic initiative designed to equip European SMEs with robust, user-friendly, and freely available tools to manage open source compliance and meet regulatory expectations. The project aims to significantly lower the costs for CRA compliance by automating key processes such as Software Bill of Materials (SBOM) generation, license verification, and vulnerability management.

This report provides a comprehensive update on the project's rationale, objectives, technical work packages, SME-oriented use cases, ongoing development, and future deliverables. It highlights all partners' contributions with this key technical milestone, and shows the collaborative synergies in delivering on OCCTET's mission: to make open source compliance and cybersecurity achievable, scalable, and sustainable for all European SMEs.

2.1 What are SBOMs

As Open Source Software is a regular component of software, from consumer goods to security-relevant systems at the national level, the Software Bill of Materials (SBOM) has established itself as a key tool for meeting legal and security requirements.[1] Similar to the classic bill of materials in the manufacturing industry, the SBOM documents all components – especially third-party components of a software application in a structured form.[2] The SBOM is typically a directory of software components that can be read by both humans and machines, including associated metadata (e.g., name, version, manufacturer, and license type).

Against the backdrop of today's highly component-based software development, many products are based on a large number of open source libraries and third-party components,

[1] In both the US and the EU, SBOMs are promoted as a key security tool through policy requirements. In the US, Executive Order 14028 mandates SBOMs for government software, accompanied by NTIA standards and CISA implementation measures. In the EU, the Cyber Resilience Act provides for transparency regarding software components; on August 4, the Federal Office for Information Security (BSI) published Part 2 of Technical Guideline TR-03183 "Cyber Resilience Requirements." Software bills of materials (SBOMs) are among the key requirements of the European Cyber Resilience Act (CRA). Software bills of materials (SBOMs) are among the key requirements of the European Cyber Resilience Act (CRA). https://www.bsi.bund.de/DE/Service-Navi/Presse/Alle-Meldungen-News/Meldungen/TR-03183-2-SBOM-Anforderungen.html, 23/09/ 2025

[2] D. Riehle, "*The Software Bill of Materials*" in Computer, vol. 58, no. 04, pp. 115-120, April 2025, https://www.computer.org/csdl/magazine/co/2025/04/10938013/25mYHqTDkZO

whose development and maintenance status can change significantly over time. For example, components may become obsolete, vulnerable to security risks, functionally inadequately documented, or without active maintenance by the community. In such cases, there is an increased risk of operational failures or significant additional costs for the maintenance and further development of the software in question.[3]

Such risks are particularly relevant for regulated industries (e.g., medical devices, aviation technology), for products with long operating or maintenance cycles (such as industrial control systems or safety-critical infrastructures), and in areas with special requirements for reliability and traceability. In these contexts, unstable or no longer available components can significantly impair compliance with industry-specific regulations, operational safety, and maintainability.

A continuously maintained and structured SBOM can be used to identify at an early stage which components are classified as risky in this respect. This makes it possible to proactively implement countermeasures such as refactoring, component replacement, or support contracts, thereby ensuring the long-term operability of the software product.

SBOMs serve three primary legal and operational purposes that are now widely recognized across both public procurement and private sector software development:

2.1.1 License Compliance

An SBOM provides the foundation for determining which licenses apply to a software product's components. This is critical for satisfying attribution obligations, fulfilling source code access requirements under copyleft licenses, and avoiding inadvertent license violations. As Heather Meeker emphasizes, tracking open source usage is essential to mitigate copyright risks and provide required notices to downstream users. This is also codified in ISO/IEC 5230:2020 (OpenChain Specification) which mandates a clearly defined process for identifying OSS components and their license obligations.

2.1.2 Security and Vulnerability Management

The modern SBOM has expanded beyond compliance to address operational and national security concerns. As Dirk Riehle notes, "to manage operational risk, you need to understand what components are doing their job… and whether new vulnerabilities have been discovered".2 This shift is also reflected in regulatory instruments like the U.S. Executive Order 14028 and the CRA , which make SBOMs a legal requirement for vulnerability tracking and response. To complement this, the ISO/IEC DIS 18974:2023 standard specifies the role of SBOMs as an integral part of a structured, security-oriented software development and maintenance process and as a basis for performing risk analyses and implementing effective measures within the framework of the Secure Software Development Life Cycle (SSDLC).

2.1.3 Supply Chain Transparency and Due Diligence

SBOMs now feature prominently in commercial contractual diligence, such as vendor assessments and M&A transactions. Customers frequently require SBOMs as part of software procurement contracts, particularly to review license compliance or identify potential exposure to restrictive terms

[3] OWASP Top 10 Risks for Open Source Software
https://owasp.org/www-project-open-source-software-top-10/ accessed on 23/09/2025

(e.g., GPL copyleft clauses). Meeker emphasizes that due diligence must include audits of embedded third-party code and license documentation. OpenChain's compliance framework reflects this in its emphasis on supplier-side transparency and documentation for inbound and outbound OSS use.

Moreover, due to the sheer scale of indirect dependencies (often comprising over 90% of the total code base) , most risk lies in components developers may not even be aware of. This reinforces the need for robust SBOM practices throughout the development and delivery lifecycle.

2.2 SBOM and CRA

The CRA requires hardware and software manufacturers to implement a series of cybersecurity measures based on the principle of "security by design." This principle requires that security aspects be taken into account at the earliest stages of product development. In addition, manufacturers are required to provide security updates throughout the entire life cycle of a product and to remedy known vulnerabilities promptly and effectively. Violation of these obligations can result in significant legal consequences, including fines, recalls, or restrictions on market access within the EU. Article 3(39) of the CRA introduces a legally binding definition of the term "software bill of materials" (SBOM): "Software bill of materials" means a formal record of the details and supply chain relationships of the components contained in the software elements of a product with digital elements;"

This explicitly qualifies the SBOM as a legally relevant artifact necessary for fulfilling regulatory obligations in the area of cybersecurity and not merely as a best practice. According to the CRA, SBOMs are an important tool for fulfilling explicit regulatory obligations related to risk management, vulnerability handling, and update requirements. An SBOM under the CRA is a means to an end, not the end itself.

2.2.1 Beyond Security: SBOMs Must Also Address Licensing Obligations

While the CRA emphasizes the role of SBOMs primarily in the context of vulnerability tracking, risk mitigation, and security updates, this focus reflects only one dimension of SBOM utility. In practice, a complete and correct SBOM also captures the legal metadata associated with each component and most importantly, the license(s) under which it is used.

This broader interpretation is not only rooted in industry practice (as defined by standards such as SPDX, CycloneDX, and OpenChain ISO/IEC 5230) but is also essential to fulfill other legal and contractual obligations, including:

- License attribution and notice compliance
- Copyleft license obligations, including source code disclosure triggers
- Commercial and IP due diligence in vendor, customer, and M&A contexts

In short, license compliance and vulnerability management are two sides of the same SBOM coin.

From a compliance and operational standpoint, this means that organizations preparing SBOMs for CRA purposes, to meet security-related obligations, should not treat license data as optional or secondary. On the contrary, failing to track license obligations in the SBOM risks exposing the organization to legal liability, especially in complex supply chains or regulated industries.

SBOMs under the CRA are essential for:

- Ensuring compliance with security update obligations

- Supporting security vulnerability reporting (CRA Art. 14) and lifecycle risk management

An SBOM fulfills several interrelated functions in the context of CRA compliance:

2.2.2 Supporting security updates and patch management

The CRA requires manufacturers to provide security updates throughout the entire period during which a product is marketed or supported. A complete and accurate SBOM enables development and security teams to identify precisely those components that need to be updated or patched as soon as new vulnerabilities become publicly known, especially in complex dependency chains.

2.2.3 Compliance with vulnerability disclosure requirements

Articles 13 and 14 of the CRA require manufacturers to establish procedures for coordinated vulnerability disclosure. An SBOM improves the manufacturer's ability to respond to reported vulnerabilities in a legally compliant and efficient manner by clearly identifying which components may be affected and whether affected software versions have already been shipped.

2.2.4 Enabling lifecycle risk management

Annex I Part II of the CRA establishes a direct link between SBOMs and risk management procedures. Manufacturers are required to continuously identify security risks, assess them, and implement appropriate risk mitigation measures. The SBOM provides the factual basis for risk analysis by disclosing exactly which software components including open source and transitive dependencies have been used.

However, an SBOM alone is not sufficient to meet the requirements of the CRA. The regulation requires active risk management, which includes the following steps:

- Knowledge of the components used
- Identifying known vulnerabilities in these components,
- Implementation of appropriate remedial measures.

Risk management therefore requires transparency about software components. Only those who know which components are contained in the software can assess the extent to which known security vulnerabilities pose a concrete threat. While this is often traceable in commercial third-party libraries through contractually regulated license and update conditions, dealing with open-source components requires proactive and documented recording. This documentation is provided by the SBOM and thus forms the backbone of all further compliance measures.

2.2.5 VEX – supplement to the SBOM in the context of cybersecurity and ICT risk management

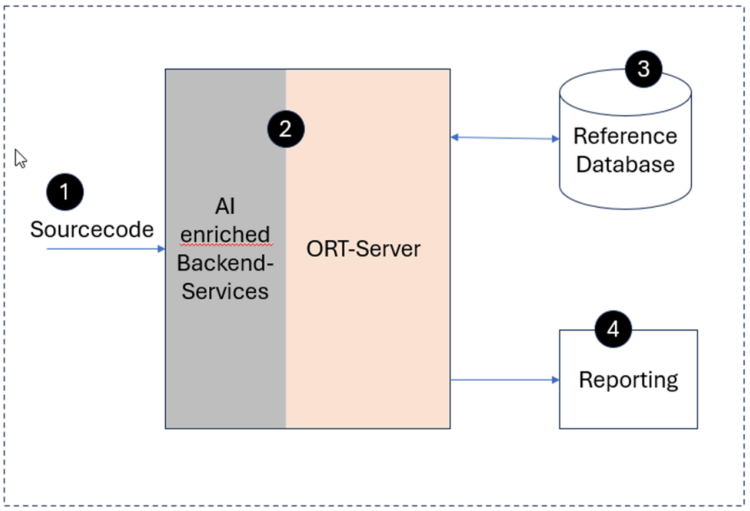
The SBOM provides a structured list of all components of a software product. It provides transparency about the components used, but does not provide any immediate information about whether known vulnerabilities in these components can actually be exploited in the respective application or operating environment. This is where the Vulnerability Exploitability eXchange (VEX) specification comes into play. A VEX document supplements the SBOM

with machine-readable information on whether certain vulnerabilities affect the product in its actual delivered state. This enables manufacturers, users, and operators to set priorities for security measures in a targeted manner while avoiding unnecessary or risk-free patches. The integration of VEX documents into SBOMs has established itself as a best practice and is supported internationally by various stakeholders. The machine-readable linking of component information with the exploitability status of known vulnerabilities contributes significantly to increased efficiency and automation in security-related processes.

The US Cybersecurity and Infrastructure Security Agency (CISA) has emphasized the importance of VEX integration in several publications, particularly for promoting the automated and effective use of SBOMs. A study published in 2025 also shows that adding vulnerability status in VEX format to SBOMs significantly increases the practical usefulness of these tools, especially for maintenance and security teams. While the authors cite challenges in standardization and automation, they also demonstrate the potential of VEX for continuous, machine-readable tracking of security risks22

3 Toolkit Overview

The following graphic shows a high-level view of how the tools involved work together.

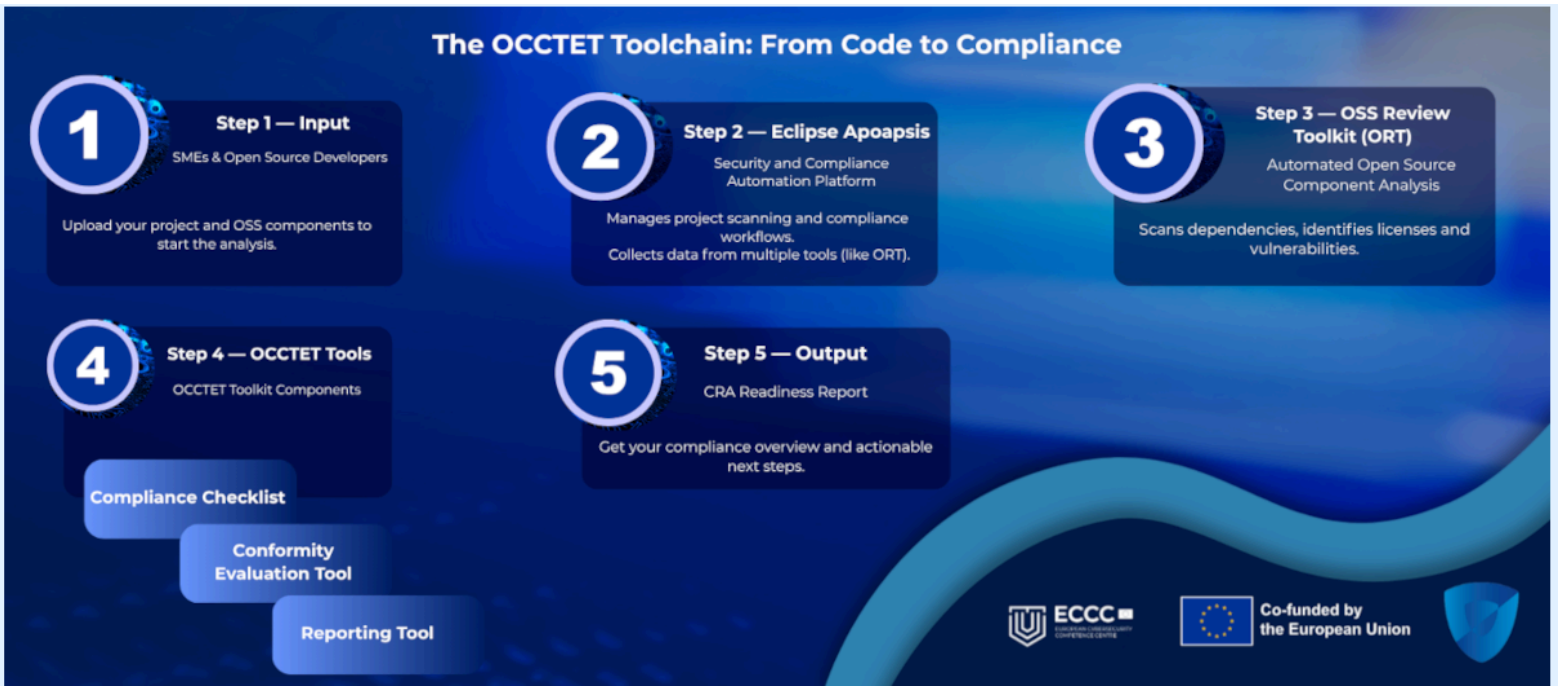


1. Sourcecode as input, normally via URL (Git)
2. ORT Server performs a code analysis of the source code, supported by AI services. An auditor must manually review the result and curate it if necessary.
3. Reference DB for Metadata and Vulnerabilities
4. Reporting-Engine, generate SBOM and Vulnerability report

Occtet Solution

Each tool is presented in detail in the following sections, including its specific function, its integration points within the toolchain, and its technological foundations.

Find a step wise application of the toolkit:



4 ORT Toolchain

4.1 The Role of Double Open

Double Open Oy is a Finnish startup founded in the year 2023 that emerged out of a research and automation initiative by the Finnish law firm HH Partners which is engaged with the OpenChain Project. Double Open strives for creating open solutions for highly automated software compliance checks and for process optimization. It does so via a unique combination of both legal and technical expertise in the open source domain, as the founders are top-tier technology attorneys and long-standing open source project maintainers.

Double Open's main product is the Double Open Compliance (DOC) solution. It is fully open source, building on and extending existing open source projects like the OSS Review Toolkit (ORT), the ORT Server, and the Double Open Server (DOS). With the ORT ecosystem being central to Double Open's offerings, Double Open (co-)maintains these projects with an "upstream first" principle. Since the beginning of the OCCTET project, Double Open has publicly contributed more than 1700 changes and improvements to the wider ORT ecosystem. In particular, Double Open developed the whole Graphical User Interface (GUI) for the ORT Server in the form of a web frontend powered by the React framework.

For the context of this milestone, the deliverable consists of the following releases:

- ORT 71.0.1
- ORT Server 0.39.0
- DOS "occtet-d3.2"

All the tools from all tasks can be called sequentially or in parallel to demonstrate basic integration and more. The tools are released as open source and can be downloaded and used by anyone.

4.2 Interactions between Tools

Historically, the core ORT project is a suite of tools implemented as library functions that are controlled by a Command Line Interface (CLI), allowing very flexible implementations of software compliance checks into a local developer's workflow and Continuous Integration (CI) workflows alike. Configuration can be shared, version-controlled, and reviewed as Yeat-Another-Markup-Language (YAML) files via Configuration-as-Code (Cac) principles. However, ORT itself aims to be unopinionated and does not provide any legal advice as such, and hence ships without any configuration.

While powerful, this flexibility comes at the cost of complexity. Less technical users who are commonly involved in compliance checks usually do not want to write YAML files, but prefer a Graphical User Interface (GUI). This is one of the aspects that the ORT Server is

addressing. It uses ORT's functionality programmatically, swapping out the CLI with a REST API and a web-frontend that makes use of that API. The server-based approach further allows to implement several features that were not possible to implement before:

- Scalability: Decouple from limitations of CI pipelines to run jobs in parallel.
- Database: Use a central database for storing and interchanging results rather than files.
- Credentials: Proper credential management to authenticate against third-party services.
- Role management: Account for different target audiences, like developers or product managers, using the tool.

ORT Server can be deployed in a managed Kubernetes (K8s) environment with several infrastructure components (like a database, messaging system, and secrets management) installed.

The Double Open Server (DOS) complements the ORT ecosystem with a more intelligent and performant license scanning solution on the one hand, as well as a GUI for reviewing and clearing license findings on the other hand.

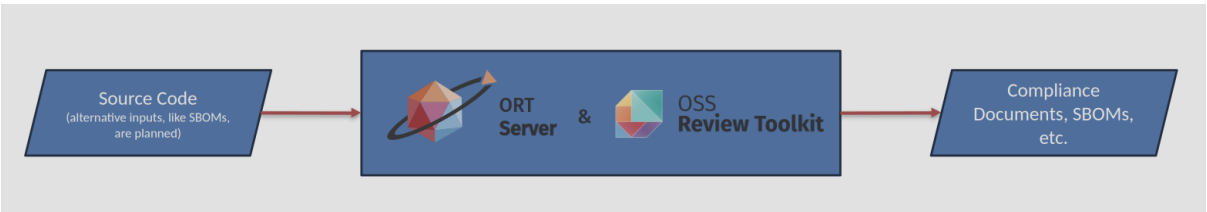
Finally, the Double Open Compliance (DOC) combines the above three tools to a holistic solution to manage end-to-end compliance processes. Not only does it come with proper technical configuration for all tools for maximum interoperability and performance, but also with sound legal configuration regarding license classifications and policy rules.

As deploying an ORT server in K8s can be challenging, DOC is also available as a Software-as-a-Service (SaaS) solution, that allows Small-to-Medium Enterprises (SMEs) to get started quickly with the solution, without the need for a significant ramp-up time.

All in all, DOC can be thought of as an enterprise-ready distribution of ORT, ORT Server, and DOS. This is similar to other enterprise-distributions of Open Source tools, like for example Red Hat Enterprise Linux (Fedora) or CloudBees CI (Jenkins).

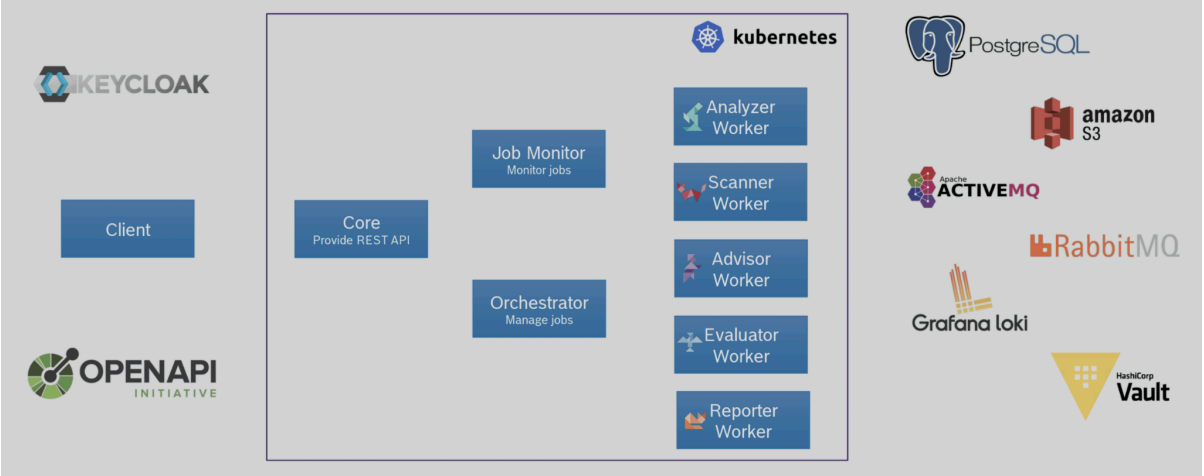
4.3 Architecture Overview

At a very high level, the compliance process implemented by DOC can be depicted as follows:



The input to the process usually is the source code of the product to conduct the compliance process for. Then ORT Server with the help of ORT orchestrates project analysis, vulnerability advice, license scanning, policy evaluation, and report generation. On the output side, fully configurable compliance documents, SBOMs, or attribution documents can be generated.

Taking a deeper look at ORT Server, the next picture shows an overview of the related infrastructure services:



The client authenticates via KeyCloak to the REST API. The core operates the job monitor and orchestrator to schedule the different ORT jobs in K8s. On the right, exemplarily services for the different required infrastructure components are shown.

4.4 Purpose of ORT Tools

With ORT being the work horse of the whole compliance process automation, it is crucial to understand the functionality of the respective tools. The following describes the tools listed as part of the architecture overview in detail.

Analyzer

The Analyzer is a Software Composition Analysis (SCA) tool that identifies the dependencies of your projects, and gathers information about them, such as licenses, copyrights, and source code locations. It

- supports more than 20 package managers, including Bazel, Cargo, Gradle, Maven, npm, PIP, pnpm, Yarn, and many more.
- works out-of-the-box with most project setups, without the need for build system changes or custom plugins.

- can curate incomplete or broken metadata for software packages, either self-written or sourced from public repositories like ClearlyDefined.
- Identify dependencies by scope to easily separate build, test, and runtime dependencies.

The Analyzer is the first step in the ORT toolchain, and its output is used by all other tools. Software packages found by the Analyzer are uniquely identified by Package URL (purl).

Scanner

The Scanner integrates third-party source code scanners to gather information about licenses, copyrights, and snippets in the source code of your projects and their dependencies. It

- automatically downloads the required source code.
- stores scan results for later reuse to avoid re-scanning the same source code.
- has both built-in and configurable mappings of arbitrary licenses to SPDX license IDs.

Advisor

The Advisor integrates various vulnerability providers to gather information about known vulnerabilities in your dependencies. It

- supports several vulnerability providers, including Google's OSV and AboutCode's VulnerableCode.
- allows for creating resolutions for found vulnerabilities, e.g. if they are not exploitable in the context of the project, or if they have been addressed otherwise.

Evaluator

The Evaluator provides a scriptable rule engine to evaluate the gathered data against custom policy rules. It

- has access to all (meta-)data gathered by any previously run ORT tool, including license, copyright, and vulnerability information.
- provides an in-built rule set based on the OSADL License Compatibility Matrix.
- can make use of custom license classifications.

Reporter

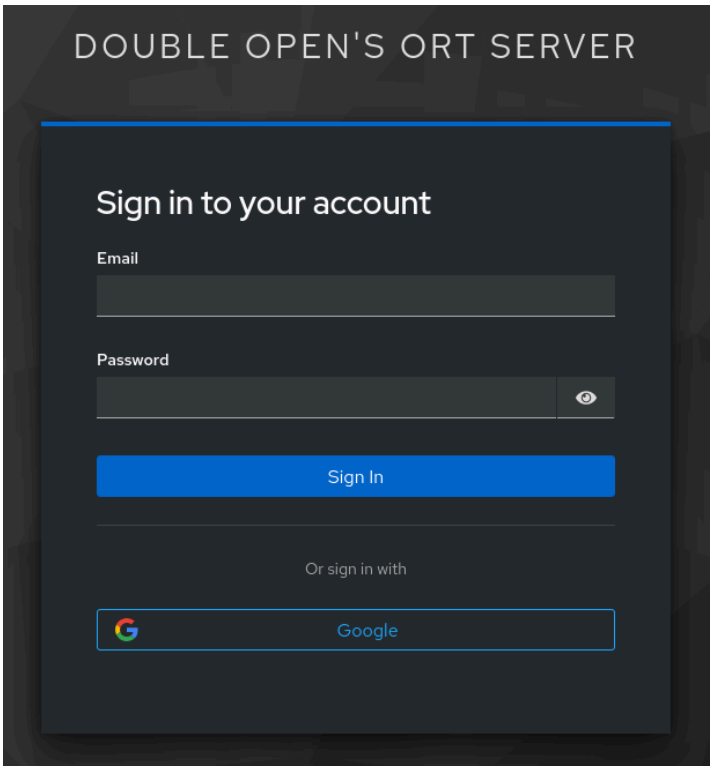
The Reporter generates various reports based on the (meta-)data gathered by the previously run ORT tool. It

- generates CycloneDX and SPDX Software Bill of Materials (SBOMs).
- provides a template reporter based on Freemarker to generate custom reports in various formats, including HTML, Markdown, or PDF.
- can build a web application in a single HTML file to visualize the gathered data.

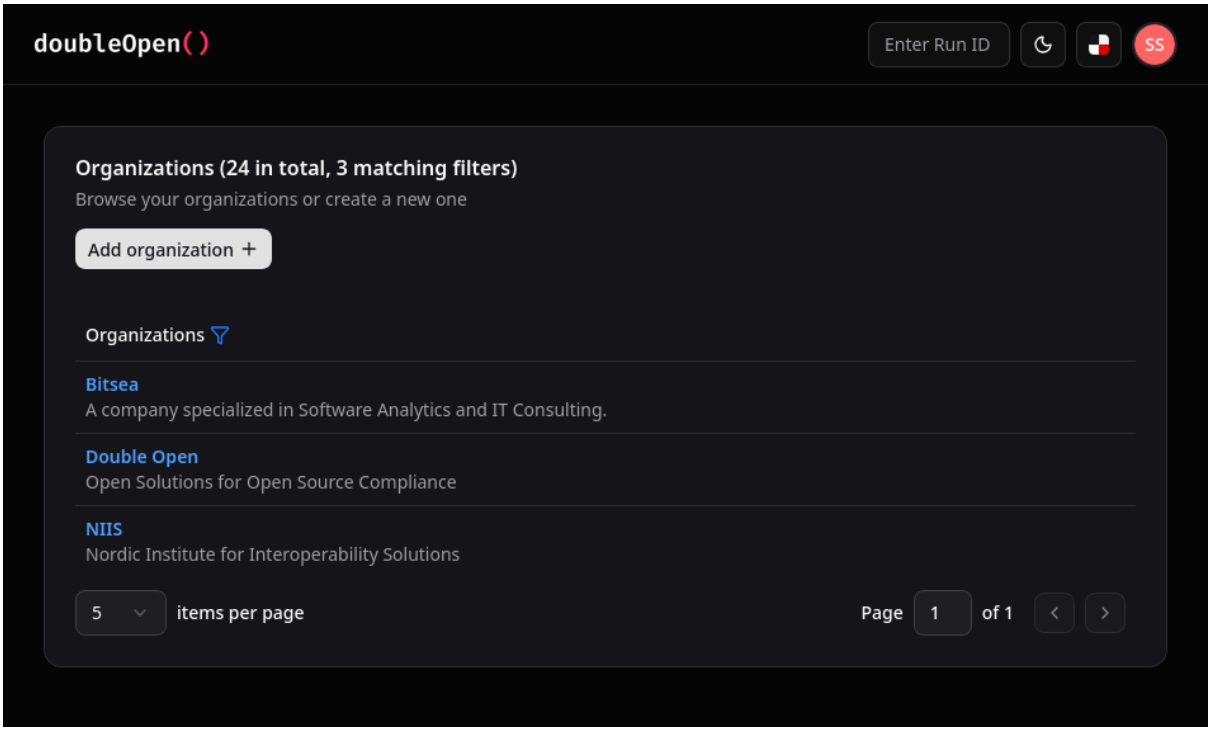
A complete user guide for ORT is available at https://oss-review-toolkit.org/ort/docs/intro.

4.5 Interaction with the ORT Server UI

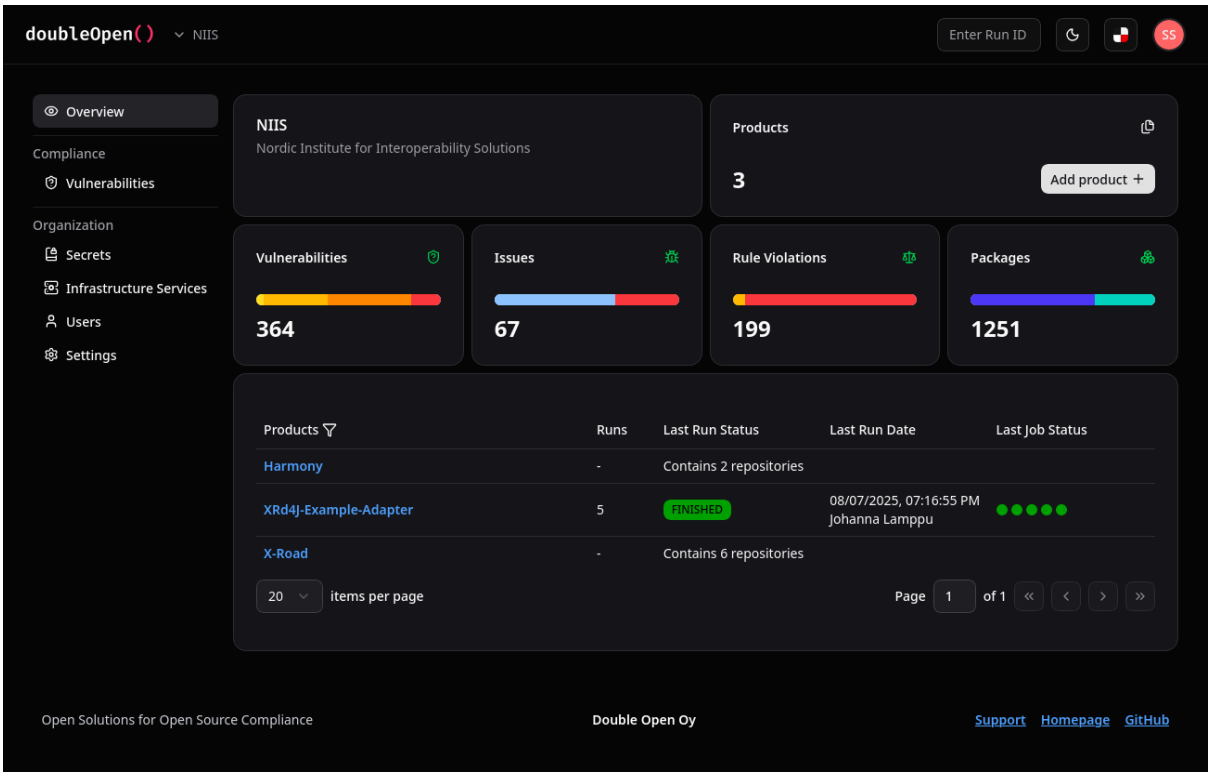
In order to use the ORT Server to create ORT runs via the UI, an administrator needs to first create an account for the user. This is done via Keycloak, either by manually creating an account, or by configuring Keycloak for OpenID Connect (OIDC), so users can use e.g. their existing Google account for authentication. See



Once logged in, the user starts at the home page which provides an overview of the accessible organizations managed in the ORT Server instance:



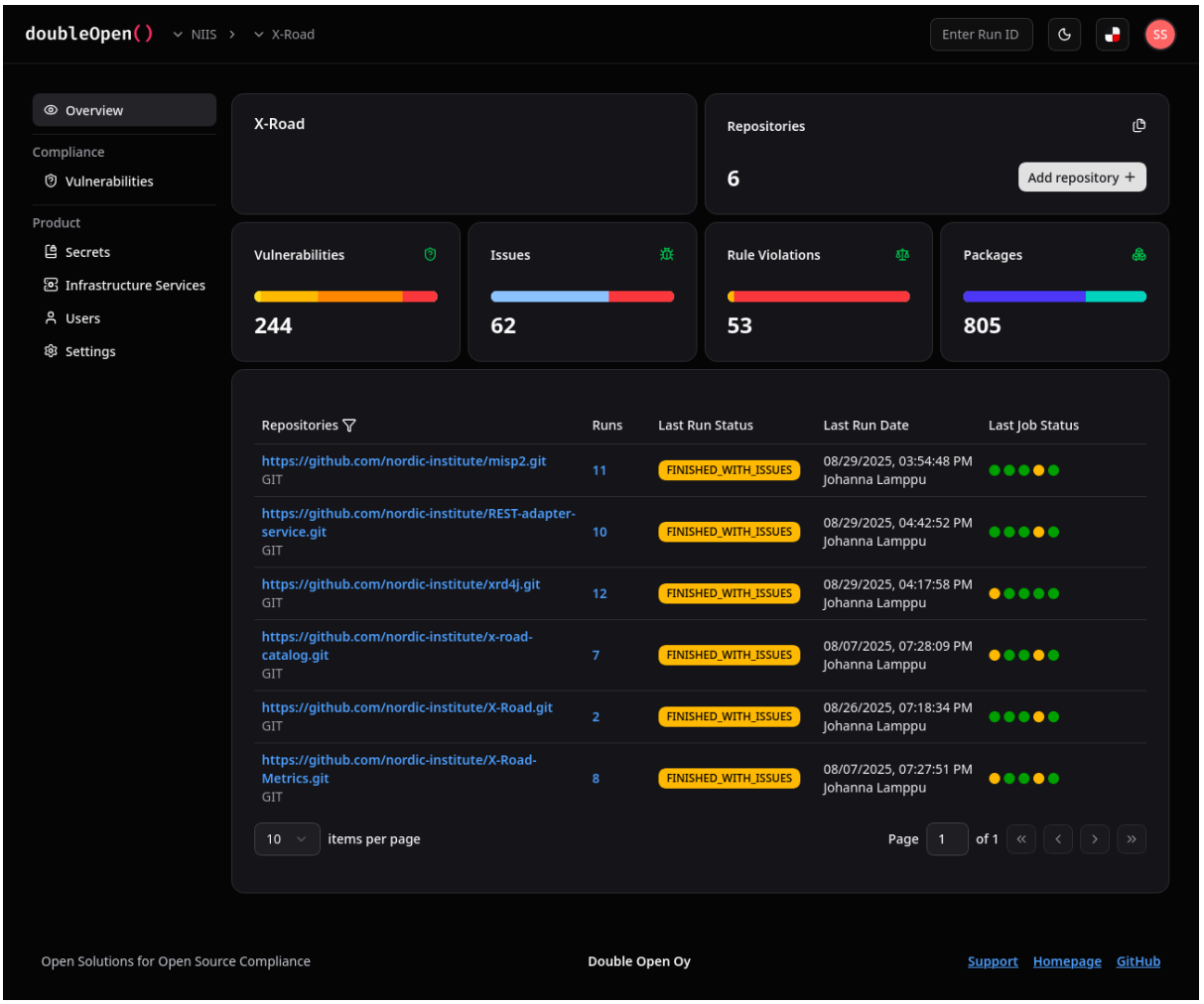
Diving into an organization by clicking e.g. on “NIIS” in the above screenshot brings the user to the product level. An organization can contain multiple products for which it wants to manage the compliance processes:



This view introduces a dashboard that providers users with the role of a product owner or similar with an organization-wide overview of

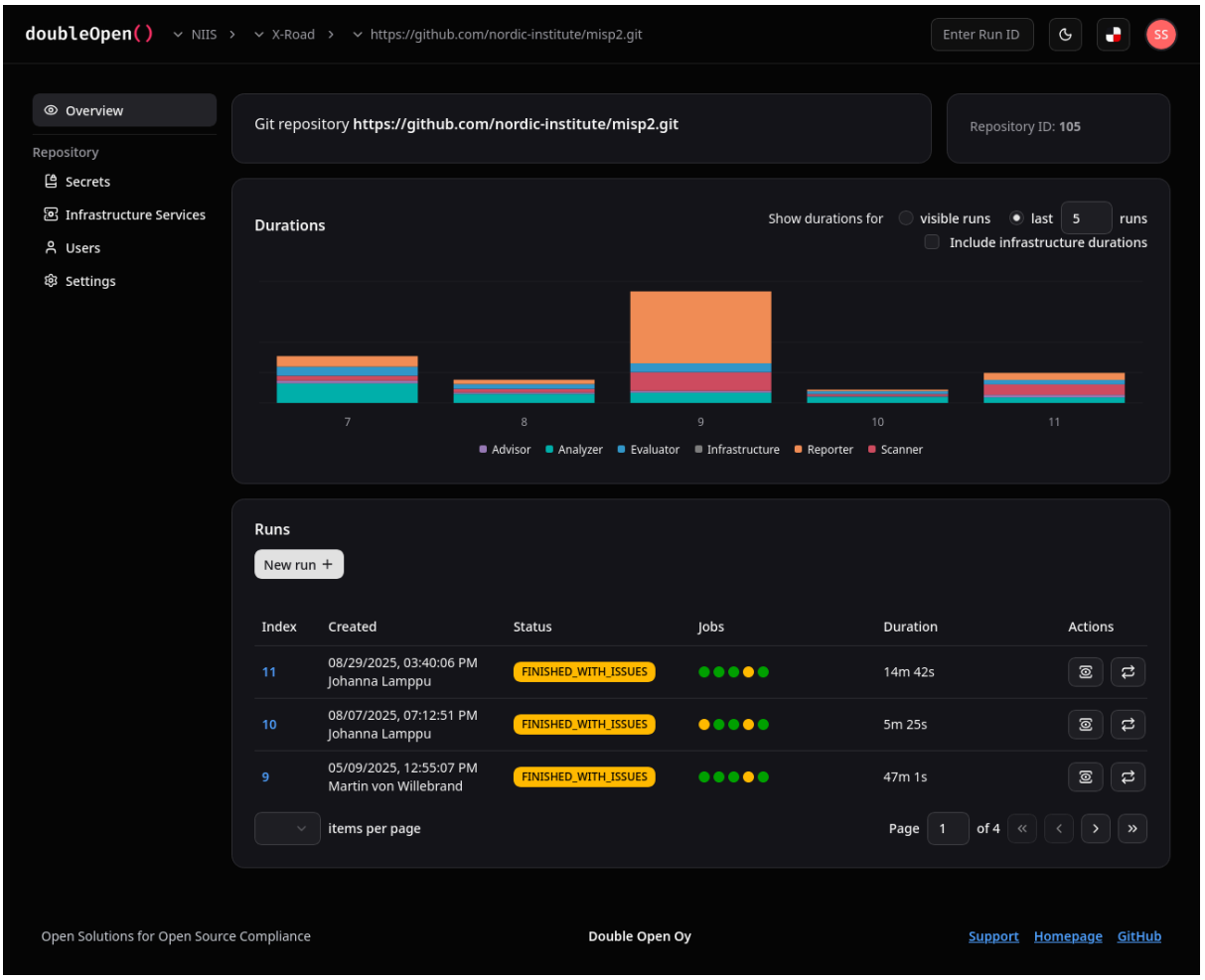
- the vulnerabilities of all products,
- technical issues of all products,
- policy rule violations of all products,
- The packages across all ecosystems of all products.

Each product in turn can consist of multiple repositories that host the source code for the product, e.g. one repository for the backend implementation and one repository for the frontend implementation of a product. At the example of the “X-Road” product, there are six repositories associated with that product:



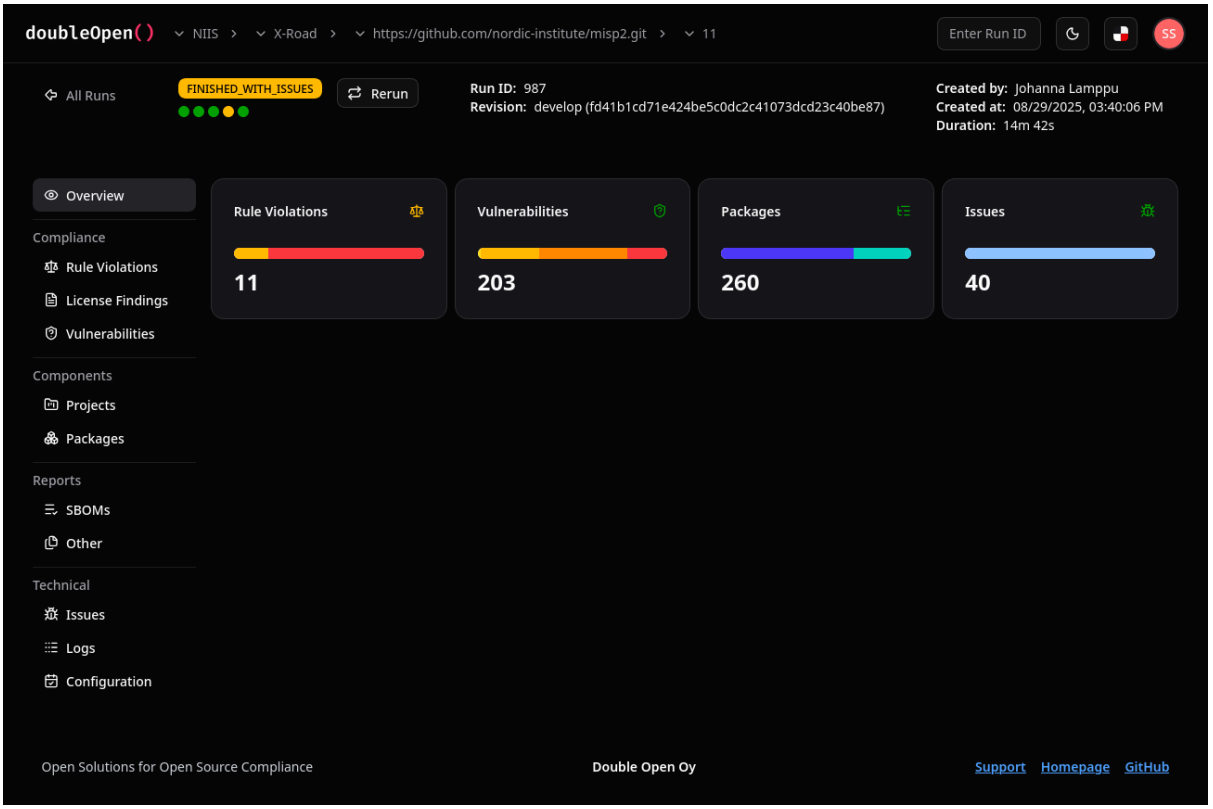
The product level view lists the repositories headed by a similar dashboard like on the organization level but scoped to the product.

The last hierarchy level is a single repository, on whose source code the ORT tools are actually run. The following is a look at all runs of the “misp2” repository:



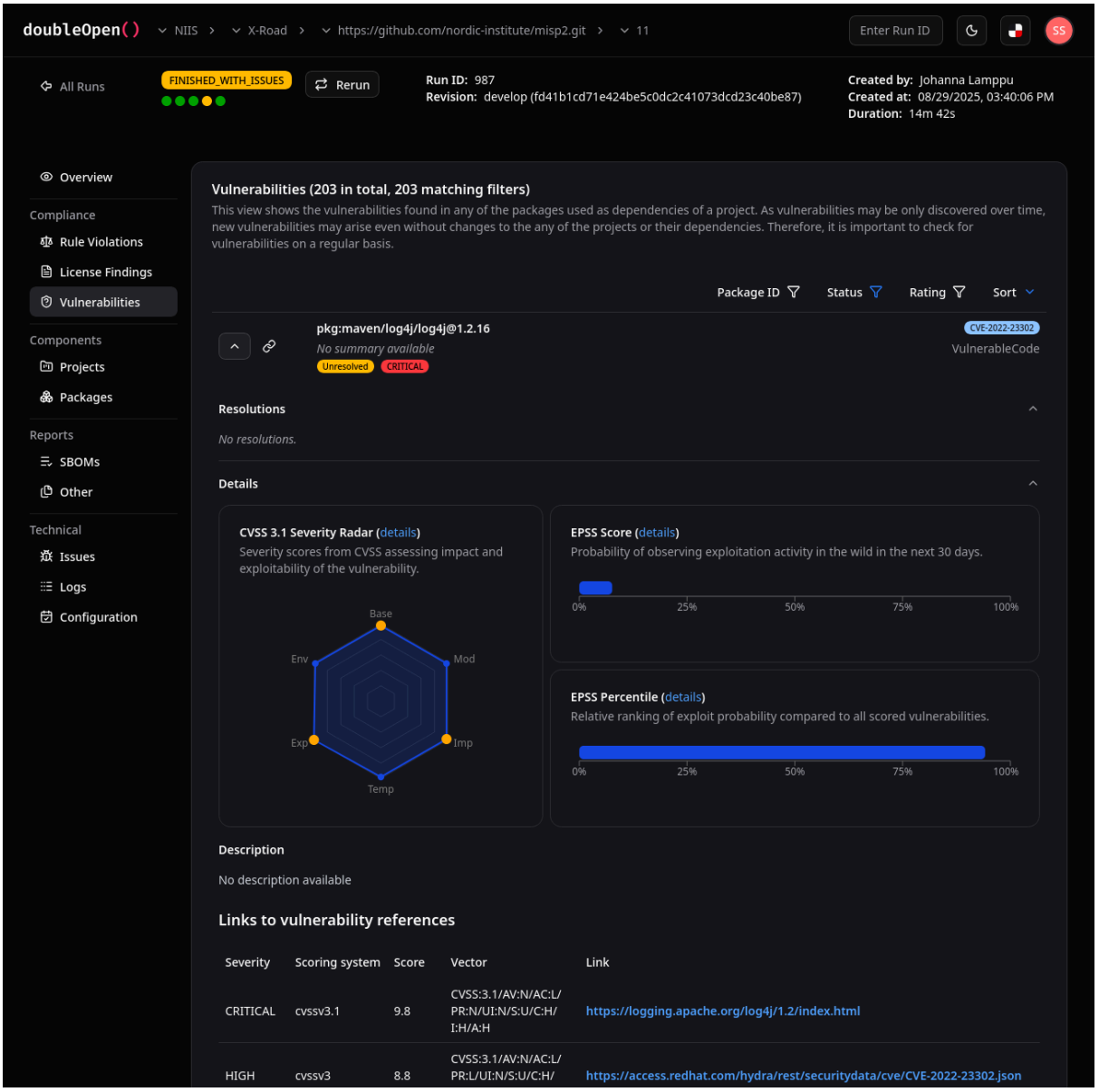
It shows the different ORT tools (Analyzer, Advisor, etc.) that were enabled for a run and their respective runtime durations, as well as the job statuses for the different runs.

The most details are available for a specific run. Picking the latest run here again takes the user to a dashboard that is now scoped to the repository and shows the number for the chosen run:



In this case, the state of the project's source code as analyzed during this run suffers from 11 policy rule violations, 203 known vulnerabilities across 260 dependency packages, and there were 40 (hint-level, as indicated by the blue-ish color) technical issues while analyzing the project. These issues could for example be hints about deprecated dependencies.

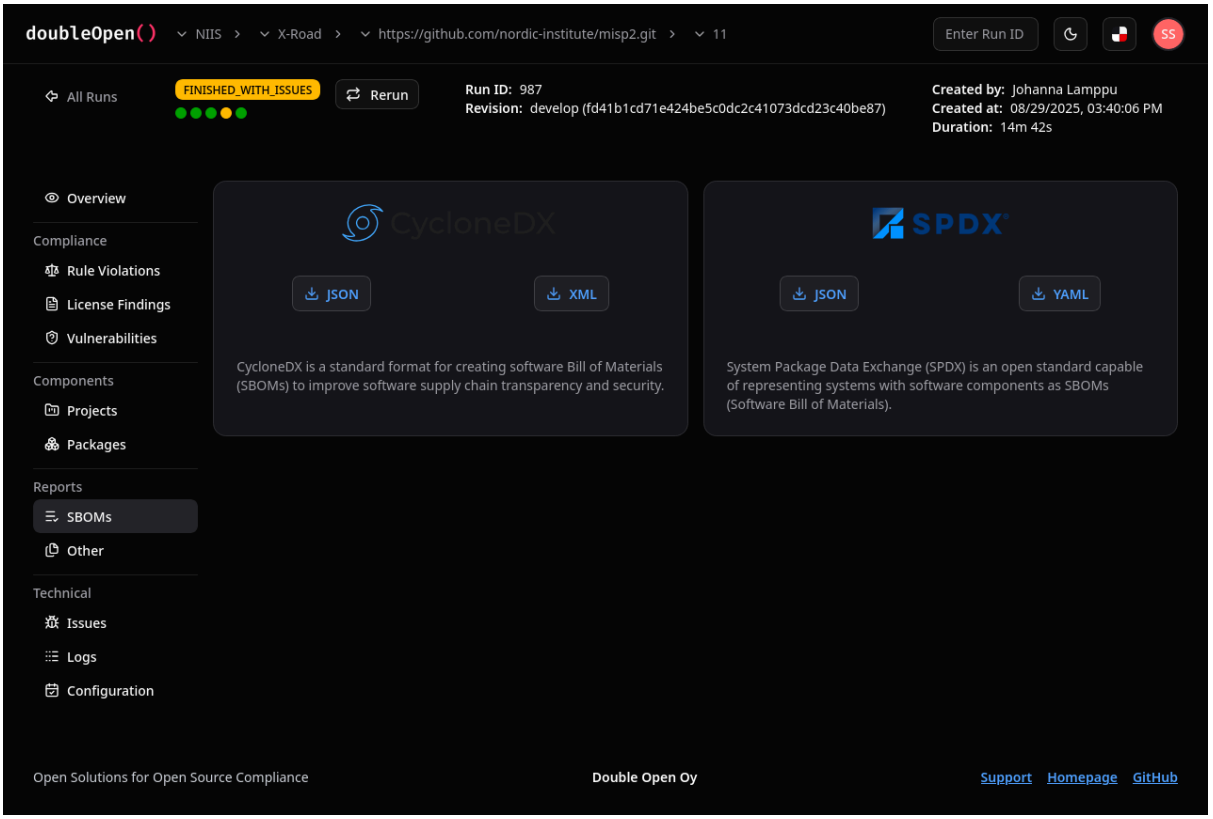
As the Cyber Resilience Act (CRA) is focussing on cybersecurity matters, the vulnerability management view is of most interest here. By clicking on the "Vulnerabilities" card, mode details about each vulnerability and references thereof become visible:



The above screenshots contains some noteworthy details about the current state of the implementation and integration between tools:

- No summary information or vulnerability description is available currently. This is due to a limitation in the integration between the ORT Advisor plugin and AboutCode's VulnerableCode provider. Double Open will resolve this problem shortly.
- No resolution for the vulnerability is available. These cannot be configured yet in the UI. This is an upcoming feature that Double Open is currently working on. Also see the roadmap section in this document.

We currently support multiple SBOM formats, and most prominently we provide SPDX and CycloneDX formats both as json and xml/yaml -files.



4.6 Envisioned End-to-End Example Workflow

The following workflow described the usage of DOC, the Double Open Compliance platform, based fully on the open source components further integrated and developed in the OCCTET project. The solution is packaged so that limited user effort is required to perform analysis. The process allows human curation and inspection of the results. It addresses the following CRA mandated and related requirements:

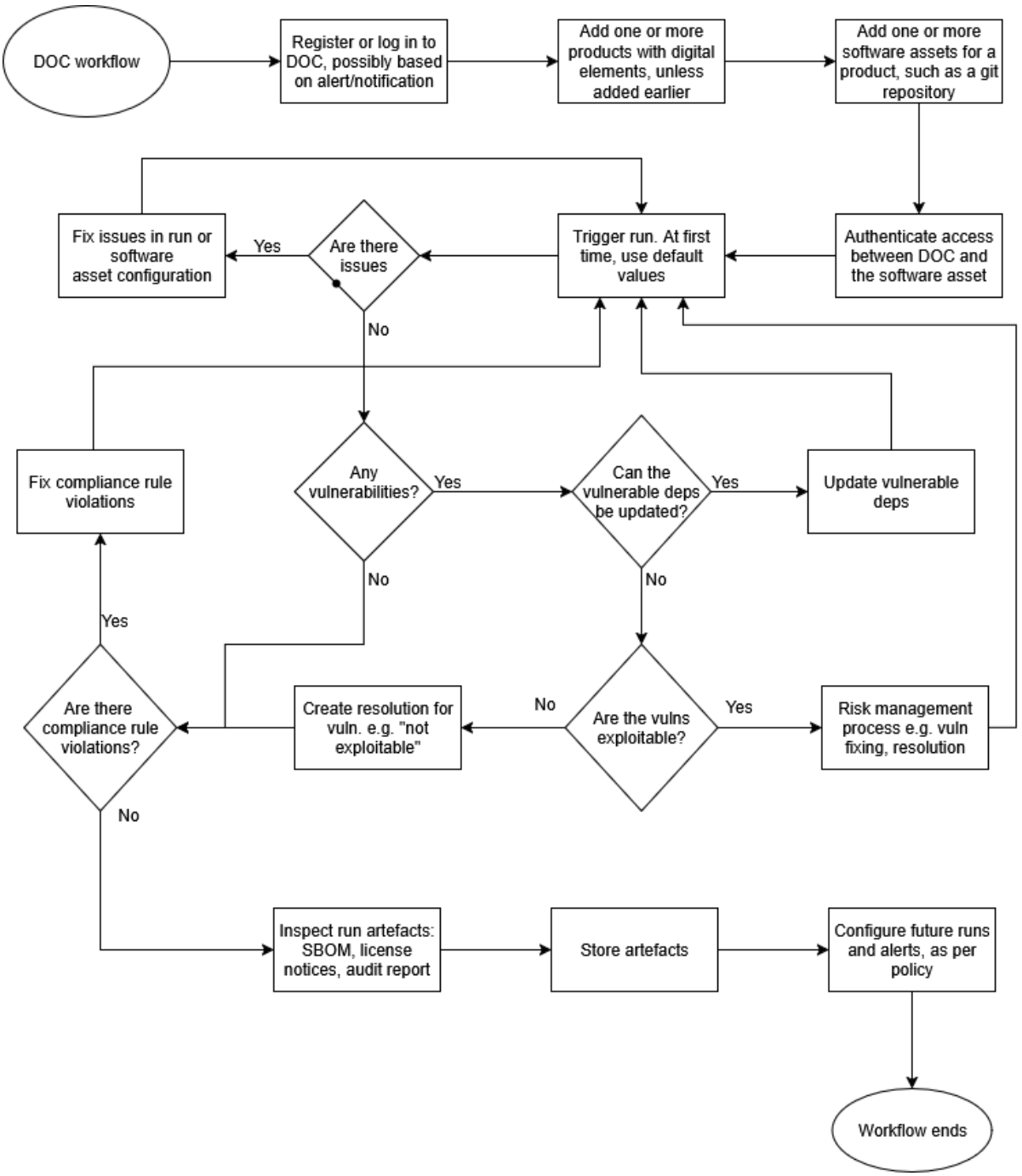
1. The fundamental vulnerability management functionality, a key requirement in CRA for all products with digital elements.
2. The SBOM creation requirement. We support among others SPDX and CycloneDX formats.
3. The creation of auditable document proving fulfillment of the above requirements and documenting the applied configuration and tooling
4. License compliance requirements, which do not flow directly from the CRA, but rather from existing legislation, such as copyright laws.

In relation to the work package 3 objectives:

1. The open source toolkit generates SBOMs in standard formats, and facilitates the efficient identification and cataloguing of software components within digital asset
2. The toolkit implements automated evaluation mechanisms to swiftly assess software vulnerability potential impact. These aspects could be further refined, considering e.g. the horizontal standards being developed by CEN/CENELEC.

3. The toolkit is planned to enable effective remediation strategies by offering actionable insights and recommendations to address software vulnerabilities through patch management, version updates, and configuration adjustments.
4. The toolkit enables assessing the security posture of open source components, exposure to known vulnerabilities, with further enhancements coming regarding the evaluating of their security practices and overall maintenance status to ensure the use of secure components in software development.
5. The toolkit automates compliance reporting with the generation of critical documents such as SBOMs and, as a future enhancement Vulnerability Exploitability Exchange (VEX) documents, aiding in adherence to Critical Risk Assessment (CRA) standards and simplifying regulatory compliance processes.

The workflow could be further enhanced for fulfilment of other CRA requirements. Those requirements will depend on the exact outcomes of the standardization work being done based on the CRA. In particular we are following the horizontal standards being developed by CEN/CENELEC.



4.7 Outlook and Roadmap

The short-term focus is on implementing an end-to-end workflow in the GUI to allow for CRA compliance vulnerability management. This includes an interactive review of known vulnerabilities found, creation of vulnerability resolutions, and scheduling of regular vulnerability scans even without any code changes happening to the product, as vulnerabilities might only be discovered over time.

Other than that, an important topic is to allow SBOM input instead of requiring access to the product's source code. This helps in scenarios where access to the source code is very restricted, or where third-party suppliers already provide an SBOM about what their deliverables contain.

Besides those major topics, continuous improvement of the User eXperience (UX) based on feedback from early adopters is key. With software compliance being a complex topic, and ORT being a powerful tool, it is Double Open's goal to abstract away the complexity and to guide users by designing accessible UI-based workflows.

While public roadmaps are currently in the works, at a more detailed level work is tracked with the "occtet" label in the issues of the respective upstream projects, which are maintained publicly at the following locations:

- ORT issues related to OCCTET
- ORT Server issues related to OCCTET

Additionally, these issues are synchronised and combined with OCCTET-related issues from other members of the consortium at the Eclipse Foundation project level.

5 Occtet-Curator

5.1 Bitsea's role in OCCTET

Bitsea is a leading provider of software audits and sustainable open source security, risk, and compliance management. For more than a decade, well-known companies in the automotive, telecommunications, financial services, logistics, and aerospace sectors have relied on Bitsea's expertise—whether for technical due diligence during M&A activities or for supporting secure digital transformation.

As an active member of the OpenChain project, Bitsea combines deep legal-technical know-how with collaborative, partner-based innovation. Within the OCCTET project, Bitsea is responsible for designing and implementing key functionality related to automated license curation, metadata enrichment, and audit result validation. By building on its proven experience in FOSS compliance, Bitsea contributes to making OCCTET a robust, regulatory-aligned toolkit for the entire European software ecosystem.

The Bitsea OCCTET Curator is designed as a practical, AI-supported web application to streamline open source software auditing for SMEs and larger enterprises alike. It targets both license compliance and vulnerability management, responding directly to growing demands under the Cyber Resilience Act (CRA) and broader software supply chain obligations.

5.2 Occtet Curator in Github

Occtet Curator is open source and can be downloaded at: https://github.com/Bitsea/Occtet-Curator

5.3 Current Status

The current version is a beta version and not all features have already been implemented. The connection to the Federated Database of ABCD is established and functional. Integration with the ORT server is not yet fully available.

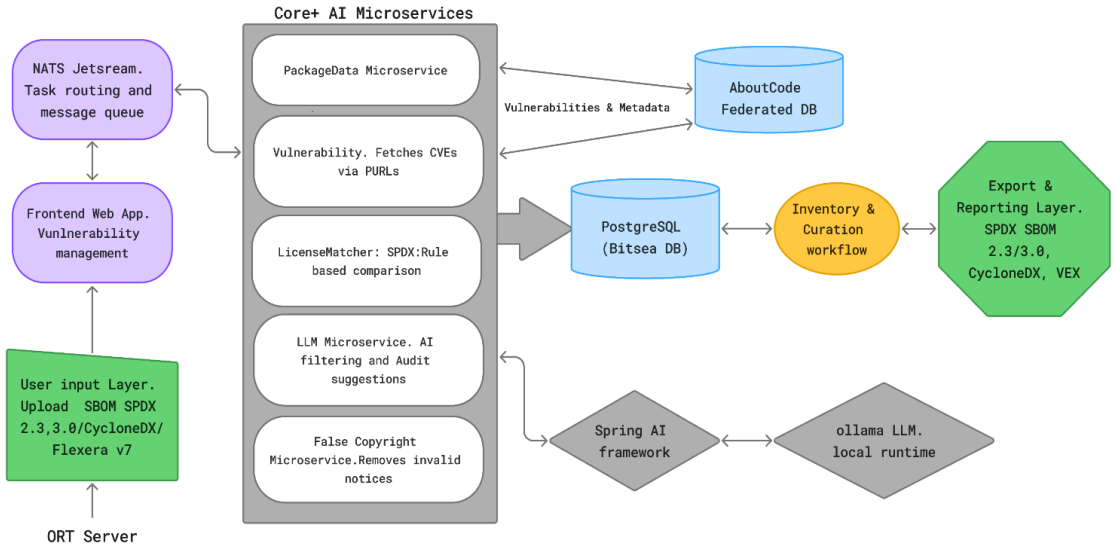
An AI is already integrated that already supports basic tasks. Basic elements of vulnerability management have been implemented.

5.4 Installation

The installation is Docker-based and described in the readme in Github. The file "`occtet-nats/docker-compose.yml`" contains all necessary configurations (ports, password, paths, mappings). The Dockerfiles are built with "`docker compose up`".

5.5 Architecture

The following diagram shows the architecture:



The Bitsea OCCTET curator is designed as a modern web-based application with a modular microservices architecture, enabling scalable, maintainable, and highly customizable audit workflows. It combines a user-friendly frontend for inventory and report management with a robust backend that orchestrates multiple specialized services. The architecture reflects key design goals of audit traceability, interoperability, and AI-assisted decision support — aligned with the needs of small and medium-sized enterprises (SMEs) working towards secure open source usage and CRA compliance.

5.5.1 User Interface

The user experience is a web-based interface built using the Jmix (v2.6) framework, which extends Spring Boot to simplify the creation of enterprise-grade applications. The UI allows users to upload SBOMs and audit reports, configure scan parameters, view results, and manage open source software inventory items and their licenses and vulnerabilities.

Each task initiated in the frontend — such as uploading a Flexera FossReport or an SPDX SBOM — results in a configuration being generated and associated with a scanning job. This configuration is passed into the system via a well-defined process and can include file uploads, string-based values, or base paths, depending on the specific scan type.

The frontend also communicates asynchronously with backend services through a messaging queue system, which supports decoupled and scalable processing of long-running or computationally intensive tasks.

5.5.2 Microservices and Task Isolation

The tool follows a task-oriented microservices architecture, where each microservice is responsible for a single, clearly defined function in the audit pipeline. These microservices are implemented using Spring Boot and communicate using the NATS messaging system, which is configured in JetStream work queue mode to support reliable and distributed task execution.

When a task is initiated from the frontend, a corresponding message is published to a subject in the NATS stream. The appropriate microservice consumes this message, processes the data, and sends back results or error messages on a response subject. This pattern ensures scalability and clean separation of concerns.

Microservices include, among others:

- SPDXMicroService: Processes SPDX 2.3 SBOMs and extracts license, copyright, and component data.
- VulnerabilityMicroService: Looks up CVE data from external sources based on software component identifiers (e.g., PURLs).
- DownloadMicroServices: Allow generation of new SPDX SBOMs or FossReports from curated inventory.
- LLMMicroService: Uses local LLMs (via Ollama) for intelligent text analysis (e.g., false positive detection, license classification)
- PackageDataMicroService: Enriches metadata for packages via external registries (e.g., PURL-based)
- DownloadSPDX/FossReport: Generates new curated SBOMs
- ControlSPDXMicroService: Validates SBOM structure and consistency
- InformationFileMicroService: Processes custom information files for AI indexing and reference

Each microservice interacts with a common PostgreSQL database (v16.0), where curated audit entities (Vulnerabilities, InventoryItem, SoftwareComponent, License, Copyright, etc.) are stored.

5.5.3 Messaging Infrastructure: NATS and JetStream

The system uses NATS as its core messaging and streaming system. NATS is a CNCF-certified lightweight publish-subscribe system, which allows various services to communicate asynchronously without direct dependencies.

JetStream, NATS' persistence layer, is activated to support the work queue model, where each task is distributed reliably and durably to the appropriate microservice. Messages are stored in a dedicated location and can be pulled by multiple clients. While currently each task queue has a single consumer, the architecture supports load balancing and parallelism in future deployments.

This message-driven architecture ensures robust handling of task status, error propagation, and result delivery, while simplifying orchestration across services.

5.5.4 AI and Local Language Model Integration

To support intelligent automation in audit workflows, Bitsea OCCTET Curator includes a dedicated AI microservice, which connects to locally hosted open source large language models via the Ollama framework. This allows sensitive code or information to be processed without relying on external cloud APIs — preserving data confidentiality and enabling offline deployment.

The AI microservice uses Spring AI for local model access and inference, and supports modular integration of additional tools such as:

- Autocomplete and text suggestion for curation fields,
- Copyright validation and filtering,

- License classification and deviation detection.

The system is also designed to support hybrid AI workflows, combining rule-based tools with retrieval-augmented generation (RAG) pipelines. These pipelines use document chunking, vector databases, and reranking techniques to improve the quality and relevance of AI responses.

5.5.5 External Data Sources and Enrichment

Several microservices are responsible for augmenting internal data with external metadata and vulnerability information. This includes:

- Querying third-party metadata databases like AboutCode, ClearlyDefined, and deps.dev,
- Fetching SBOM fragments and license texts from GitHub or SPDX.org,
- Contacting vulnerability databases via CVE or PURL identifiers.

The system is designed to support enrichment-on-demand, reducing the need for complete upfront data and supporting incremental curation.

5.5.6 Open Source Licensing and Sustainability

Bitsea OCCTET Curator is released as free and open source software under the Apache-2.0 license. The source code is publicly available on GitHub, allowing other audit providers, security teams, or compliance officers to reuse and adapt the toolkit for their own needs.

5.5.7 Deployment and Packaging

Bitsea OCCTET Curator is designed for flexible deployment to accommodate the varying technical capacities of its users — including small and medium-sized enterprises (SMEs). It can be deployed either as:

- A self-contained JAR file for each microservice and the frontend UI, using the standard Spring Boot build process,
- A project-wide Gradle build, enabling consistent compilation, dependency resolution, and artifact generation,
- Or as a Docker image, allowing easy containerization and orchestration using Docker Compose or Kubernetes.

For users familiar with container-based environments, each component of the Bitsea OCCTET Curator architecture — from the NATS server and PostgreSQL database to the individual Spring Boot microservices — can be encapsulated in a Docker image. This enables reproducible environments, versioned deployments, and simplified dependency management, which are especially beneficial in regulated or security-sensitive settings.

Pre-built Docker images may be provided for convenience, or users can generate them via the Gradle build pipeline. This approach supports both local development setups and production deployments in cloud or on-premise infrastructure.

5.5.8 Docker images

To support portability, reproducibility, and ease of setup, OCCTET Curator will be delivered as a set of 4 to 5 Docker images, each representing a core component of the system. These

pre-built images package each microservice or service layer along with its required runtime environment, dependencies, and configuration.

The images are expected to include (tentatively)

- The Jmix-based frontend UI,
- One or more backend microservices (e.g., SPDX and FossReport processors),
- The AI microservice, which runs Spring AI integrated with locally hosted open models via Ollama,
- And optionally, utility components such as the NATS JetStream server or a database initializer.

This containerized architecture allows the tool to be deployed easily via Docker or Docker Compose, making it suitable for both local development and scalable production environments. The use of container images also facilitates compliance and reproducibility in regulated sectors.

5.5.9 Vulnerability Detection and CRA Alignment

5.5.9.1 Vulnerability Management via Software Bill of Materials (SBOM)

Bitsea OCCTET curator supports both SPDX and CycloneDX SBOM formats and uses them to detect known vulnerabilities through component identifiers (such as PURLs). This information is matched against public vulnerability databases (like GitHub Advisory DB, NVD, etc.) to assess security risks. When scanner results contain missing or inaccurate metadata (such as license name, version, or source), the AI fills in the gaps based on past curated data, reducing manual work and improving accuracy.

5.5.9.2 CRA Readiness and SBOM Quality

While the CRA emphasizes vulnerability disclosure, OCCTET ensures the SBOM also meets broader industry expectations, including full licensing, authorship, and dependency traceability. This holistic approach supports organizations seeking CRA compliance and internal software governance at the same time.

5.5.10 VEX integration

The Bitsea OCCTET Curator is designed not only to help users identify and manage vulnerabilities in open source components but also to support structured documentation of their exploitability status through VEX (Vulnerability Exploitability eXchange) integration. The Curator can generate VEX documents in formats aligned with SPDX 3.0 or CycloneDX standards, or prepare them as standalone machine-readable statements.

After all findings are curated, the project can export a new SBOM (in SPDX 2.3,3.0, CycloneDX) with validated and annotated vulnerability information. These exports fulfill the CRA requirement of maintaining accurate, up-to-date documentation of security-relevant components. Organizations can share these curated SBOMs with suppliers, customers, or regulators as part of their cybersecurity posture reporting.

5.6 Roadmap

Focus Area	Objective	Planned Measures and Enhancements
Integration & System Architecture	Deepening interoperability with the ecosystem of development tools	The **connection to the ORT Toolchain** will be substantially expanded and optimized to ensure seamless data exchange and integrated functionalities
Security & Compliance	Establishing robust processes for vulnerability management and risk mitigation	In the area of **"Vulnerability Management,"** a **standardized and documented workflow** (Incident Response and Remediation) will be established. Additionally, **comprehensive features for management** (e.g., automated prioritization, reporting functions, and CVE integration) will be implemented
Internationalization & Localization	Accessing the European market through linguistic adaptation of the user interface	The **WebApp will be made multilingual** for the **EU region**, designed and aligned to increase accessibility and user-friendliness for international users
User Experience (UX/UI)	Significant improvement of the application's intuitive operation and visual design.	The **User-Experience (UX/UI)** must be evaluated and comprehensively optimized in many **critical areas** (e.g., navigation, data visualization, onboarding) to boost user adoption and efficiency
Innovation & Performance	Expansion of intelligent functionalities and improvement of the technical foundation.	**AI Support** (Artificial Intelligence) will be significantly **expanded** through the integration of new models and use cases (e.g., for decision support or data analysis). In parallel, the overall application's **Performance** will be **increased** through technical optimizations (e.g., refactoring, loading times, scalability

6 Federated database

6.1 Aboutcode's role in OCCTET

AboutCode Europe ASBL is a non-profit established to promote, support, and sustain free and open source software projects, in particular the AboutCode open source software stack. AboutCode projects include the internationally recognized and industry leading ScanCode code scanner, the industry leading Package-URL software package identifier standard, and the VulnerableCode open code and open data aggregated and correlated software vulnerabilities database. AboutCode tools are used throughout the industry with multiple millions of monthly downloads.

The core role and contribution of AboutCode in OCCTET is the deployment of an open federated database of reference data to support supply chain operations and SBOM enrichment with a primary focus on security vulnerabilities and a secondary focus on other metadata: code origin and licenses. The secondary contribution is with AboutCode core scanning tools like ScanCode toolkit that are integrated in ORT and DoubleOpen.

6.2 Federated Database overview

The Initial FederatedCode Demonstrator released in D3.1 has been focused on software package vulnerabilities. It includes the following key components:

A. VulnerableCode which is an aggregated open-source database that tracks known security vulnerabilities and is continuously updated. It serves as a comprehensive resource for identifying vulnerabilities in software packages.

Key VulnerableCode resources for that demonstrator include:

- The code is available at: https://github.com/aboutcode-org/vulnerablecode
- The demonstrator is publicly accessible at: https://public.vulnerablecode.io/
- Documentation can be found at: https://vulnerablecode.readthedocs.io

B. Decentralized Data Repository

The data used in that demonstrator is published in a Git repository, accessible at:

https://github.com/aboutcode-data/vulnerablecode-data

C. AboutCode Hashid

This data is published using the a content-defined PURL-based key computed with the AboutCode hashid library, which can be found at:

- https://github.com/aboutcode-org/vulnerablecode/tree/main/aboutcode/hashid
- The package is also available on PyPI: https://pypi.org/project/aboutcode.hashid/

Other components for the solution include:

D. FederatedCode

The code federation layer. This is available and publicly deployed, but not yet fully operational. It is usable for basic integration in the OCCTET toolchain.

- Code at https://github.com/aboutcode-org/federatedcode
- Docs at https://federatedcode.readthedocs.io
- Demo system at https://demo.data.aboutcode.org/

E. PurlDB

The database of all package metadata keyed by PURL. This is available and publicly deployed, but not yet fu

- Code at https://github.com/aboutcode-org/purldb
- Docs at https://aboutcode.readthedocs.io/projects/PURLdb
- Demo system at https://public.purldb.io/api/

6.3 AboutCode key Federated Database developments since D3.2

AboutCode key developments since D3.2 (at Month 6) and towards D3.4 (planned for Month 17) include:

A) An extensive update to VulnerableCode data models and architecture.

There we have changed the design from being centered on a concept of Vulnerabilities to being centered on a concept of Security Advisories. Vulnerabilities were generic (e.g., a CVE) while Advisories are package-specific (e.g., the Package Foo version 1.2 is impacted by a Security Advisory ABC1 which happens to be related to a CVE).

This is a deep and profound change that was made available in a new release at https://github.com/aboutcode-org/vulnerablecode/releases/tag/v36.1.1 and new point release since.

This deployed in preview at https://public2.vulnerablecode.io/v2

With a new API at https://public2.vulnerablecode.io/api/v2/

The work on support for advisories with refinements and debugging is on-going and the V1 API and models are kept in parallel live and available until this is completed. This is to ensure that OCCTET partners have access to a stable V1 API until we switch to the improved V2 API.

In addition we have made significant progress on the migration of VulnerableCode data importers and improvers to use the ScanCode.io pipeline architecture.

https://github.com/aboutcode-org/vulnerablecode/tree/main/vulnerabilities/pipelines

B) On going PurlDB and FederatedCode development and improvements.

We released a version of PurlDB: https://github.com/aboutcode-org/purldb/releases/tag/v7.0.0

Another major development is a new design to Collect package metadata with a three-tier, progressive and decentralized approach. The tiers are:

- Package names and versions. The bulk has been completed and is being deployed progressively
- Package metadata. This has been partially available and is being refined for on-demand data collection.
- Package detailed file-level scans. This has been partially available and is being refined for federated storage.

Another development is a significant rework of the **PURL**-based hashid to address data growth opportunities. In particular, supporting a configurable hash-length using fixed hash-length for a **Federation**, and introducing the concepts of **DataFederation**, **DataCluster** and **DataRepository**.

See https://github.com/aboutcode-org/vulnerablecode/tree/747-new-hashid/aboutcode/federated for an overview on the work in progress.

All these AboutCode developments and ongoing work will be part of the next Deliverable D3.4 - FedDB-Final-Del, planned for Month 17.

7 Conclusions and next steps

The OCCTET project has successfully achieved progress in the development of the planned open-source tools for software compliance, particularly in the context of the Cyber Resilience Act (CRA). Key achievements include the beta releases of the ORT Toolchain and the Occtet-Curator, demonstrating basic integration and providing functionalities for SBOM generation, vulnerability management, and license compliance supported by the AboutCode reference data.

Next step efforts will focus on integration testing, deepening interoperability, improving vulnerability management workflows, optimizing user-friendliness, and expanding AI support to ensure the long-term utility and availability of the open-source tooling solution.

Most importantly, the consortium partners have developed an excellent and deep working relationship while collaborating since the start of the OCCTET project, and are confident in their ability to complete the project.

7.1 Developed tools in the toolchain

OSS Review Toolkit (ORT)	https://github.com/oss-review-toolkit/ort/releases/tag/71.0.1	ORT being the work horse of the whole compliance process automation
ORT Server	https://github.com/eclipse-apoapsis/ort-server/releases/tag/0.39.0	ORT Server is a scalable way to operate ORT in the cloud with a web-frontend as the GUI
Double Open Server (DOS)	https://github.com/doubleopen-project/dos/releases/tag/occtet-d3.2	ORT companion project providing intelligent scanning and a clearance GUI
Occtet-Curator	https://github.com/Bitsea/Occtet-Curator/releases/tag/BasicChainDel-D3.2	Supports vulnerability management and curation of scan results from the ORT server, generates SBOM and VEX
Federated database	https://github.com/aboutcode-org/vulnerablecode/releases/tag/v36.1.3 https://github.com/aboutcode-org/federatedcode	Aggregated open-source database that tracks known security vulnerabilities

7.2 KPI

KPI 1: Number of tools to facilitate and automate CRA compliance

As described above, three tools will be provided to support SMEs with CRA compliance.(ORT Toolchain, Octet Curator and Federated database).

KPI 2 & KPI 3: CRA essential requirements fully/partially covered

As the tools are currently untested and unfinished, no CRA requirements are fully met. Once the work is complete, all requirements will be met.

KPI 4: Tools to simplify and automate CRA documentation obligations

Once the work has been completed, Occtet Curator will be able to fully comply with documentation obligations, whereas at present it can only do so partially.

7.3 Potential Risks

Complexity of Integration: The document mentions that all tools can be called sequentially to demonstrate a basic integration. This suggests that the complete and seamless integration of all components of the toolchain could pose a challenge, especially with regard to the interactions between the tools and the orchestration of the end-to-end workflow.

Risk Mitigation: We elected to use standard SBOM formats to exchange data between tools. While this is not always the absolute best possible technical solution, this approach mitigates many of the integration issues as each technical component can reuse well defined, publicly specified data structures.

Continuous Improvement and Testing: The project is in the phase of testing the tools and incrementally improving them based on the test results. This is an ongoing process, and there is a risk that unexpected errors or performance issues may arise during testing, requiring additional development time.

Risk Mitigation: We are deploying a test bed environment at Eclipse to support extensive and advanced integration testing.

Adaptation to Changing Regulations: The development must closely align with the Conformity Assessment Specifications and the requirements of the Cyber Resilience Act (CRA). Changes in these regulations could necessitate adjustments to the toolchain, posing a risk to the timeline and resources.

Risk Mitigation: We are engaging with other open source community organizations like the OpenRegulatoryComliance working group and several experts directly involved with the standardization of the CRA processes and deliverables at ETSI, and CEN-CENELEC. We are confident that any refinements in the CRA deployment will not affect core requirements such as creating and ingesting SBOMs, performing continuous software inventory tracking and completing effective and efficient vulnerability management. These are core functional areas that are essential to the CRA and will stay core.

Quality and Correctness of SBOMs: Although the goal is to create accurate SBOMs, the implementation of an AI/ML-supported review of scan results and SBOMs for correctness is also mentioned. This implies that ensuring the high quality and accuracy of the generated SBOMs can be a challenge and requires continuous validation.

Risk Mitigation: Our whole approach that combines open tools and open data is designed to improve and ensure a better correctness of SBOM content. The situation of SBOM correctness and quality is an emerging issue across the SCA and regulatory compliance space. This is a key issue we are addressing with our toolchain and data. We are proactively engaging with key initiatives in that domain including with the OpenChain project Reference tooling group, the OpenChain Telco group and Nokia, as well tracking studies from organizations like nexB and Carnegy Mellon. Overall we expect that our open toolchain will contribute significantly to improved quality and correctness of SBOMs with free and open source tools and data, readily deployable by any organization.

User Acceptance and Effort: The document emphasizes that the solution should be packaged to limit the user effort for analysis. Ensuring a user-friendly installation and operation of the beta versions of the tools is crucial for acceptance, especially among SMEs who may not have dedicated legal or compliance teams.

Risk Mitigation:

- **UX/UI Integration:** Early and continuous involvement of UX designers to create an **intuitive and minimalist interface**.
- **Simple Installation:** Creation of **all-in-one installation packages** (e.g., Docker or simple executables) for an "out-of-the-box" start.
- **Process Automation:** Maximum **automation of analyses** and reporting to minimize manual interventions (Key concept: the **"Zero-Click" Principle**).
- **Target Group Testing:** Conducting **usability tests and pilot projects** with **SMEs** (the primary target group) to gather feedback on user-friendliness.
- **Clear Documentation:** Creation of **short video tutorials** and **"Quick-Start Guides"** in plain language.
- **Results Visualization:** Outputting **prioritized, actionable recommendations** and visual status scorecards (e.g., a traffic light system) instead of just raw data.

Vulnerability Management: Task 3.3 focuses on automating triage, security posture assessment, and vulnerability remediation. This is a complex area, and developing robust tools that can effectively distinguish between actual vulnerabilities and non-exploitable cases (VEX integration) is a major challenge.

Risk Mitigation:

In summary, the main risks lie in the complexity of integration, the continuous adaptation to external factors (open-source projects, regulations), and ensuring the quality and usability of the developed tools.

We are aware of the identified risks and attach great importance to their potential impact on the project schedule. For this reason, we not only carefully and continuously evaluate these risks, but also proactively and strategically deploy targeted countermeasures.

These actions aim to reduce the probability of occurrence or the severity of the impact, thereby ensuring effective risk mitigation. Through this structured approach, we ensure that compliance with our timeline, the reliability of our technology, and ultimately the successful achievement of our overarching project goals are secured.

7.4 IPR and Licensing Notice

8 Annex 1 - Occtet-Curator Screenshots

Figure A1-1: This mask shows the set of all vulnerabilities already present in the system with some metadata.

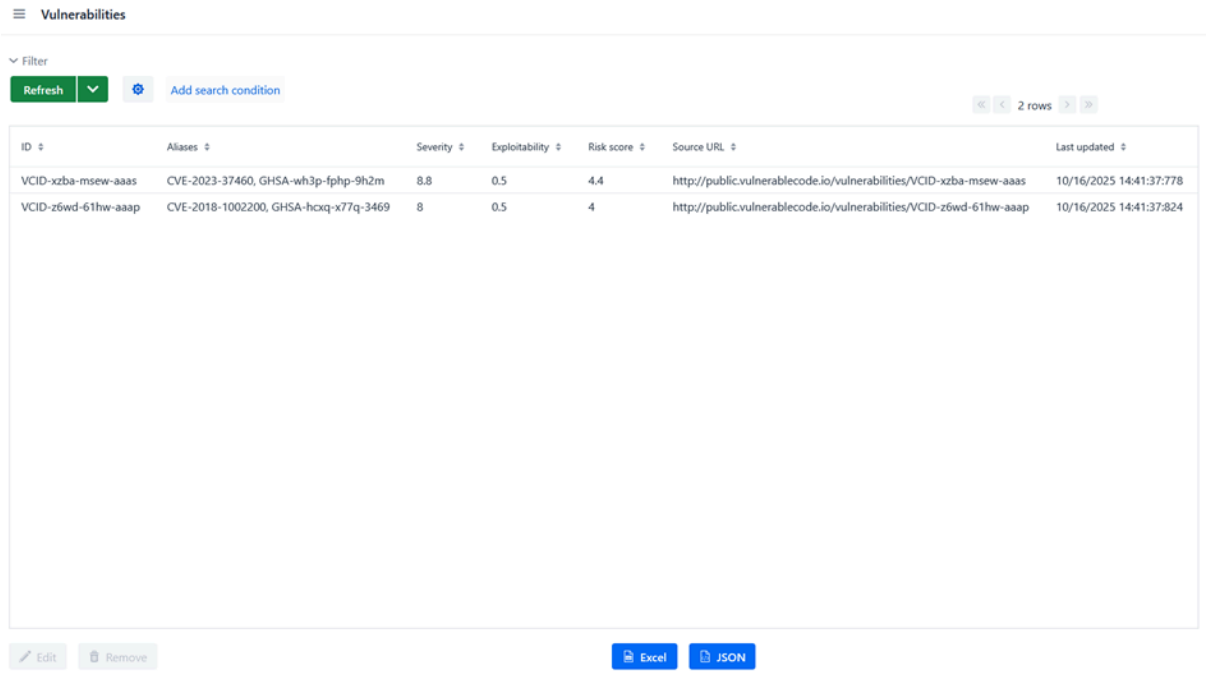


Figure A1-2: This mask shows the vulnerabilities associated with a SW component

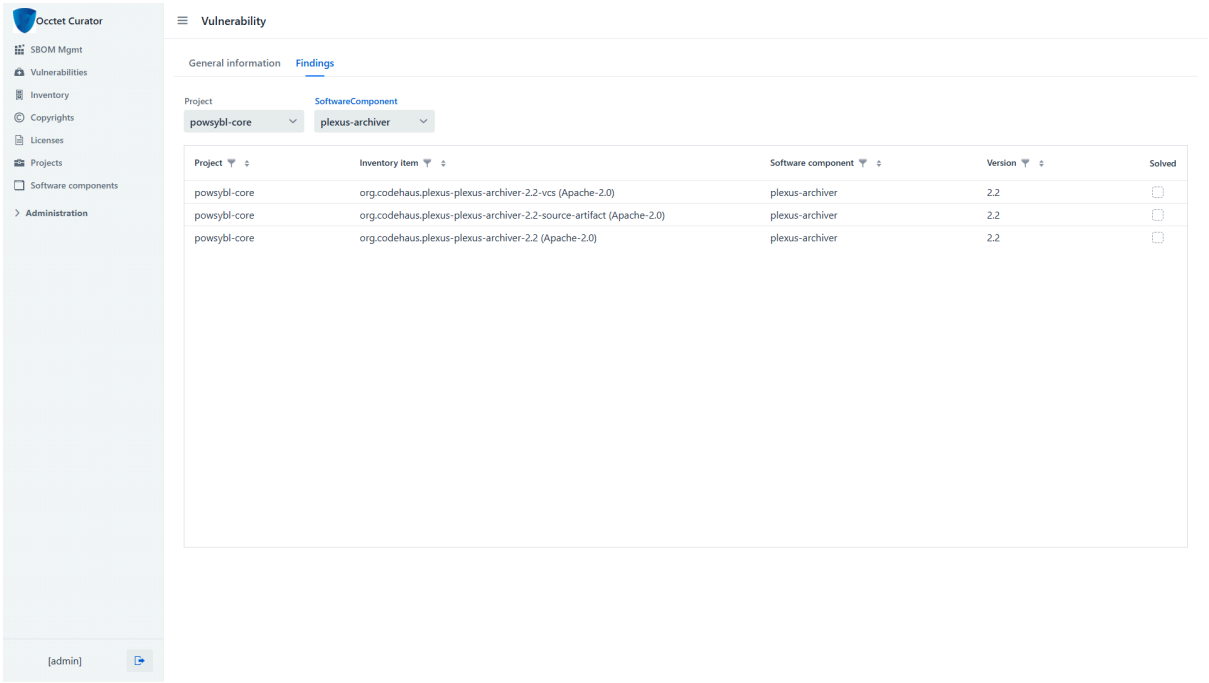


Figure A1-3: This mask shows the data of a vulnerability retrieved from the Federated Database (ABCD)

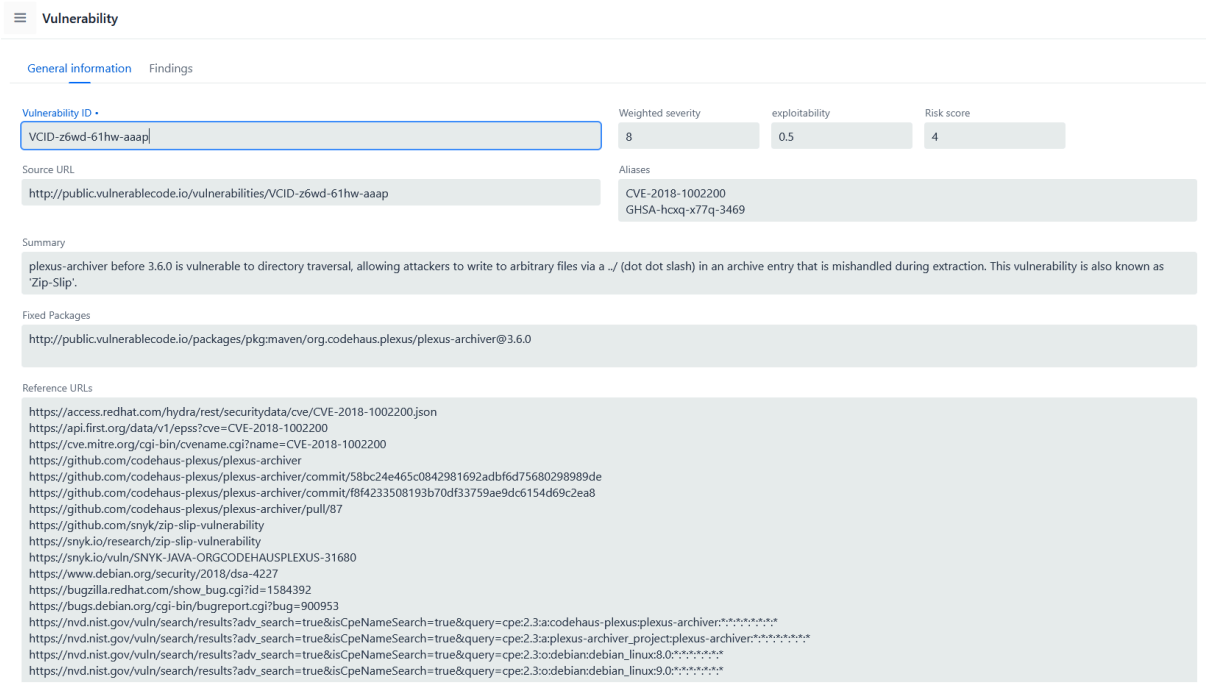


Figure A1-4: Once the vulnerabilities have been processed, a VEX can be created

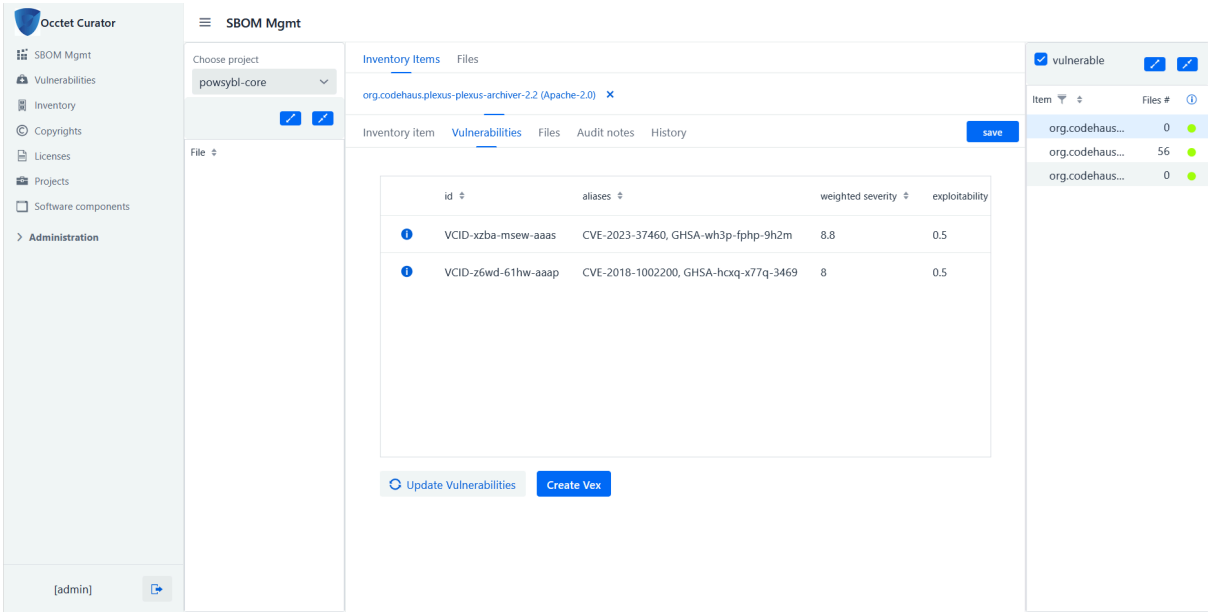


Figure A1-5: The form of a VEX edit.

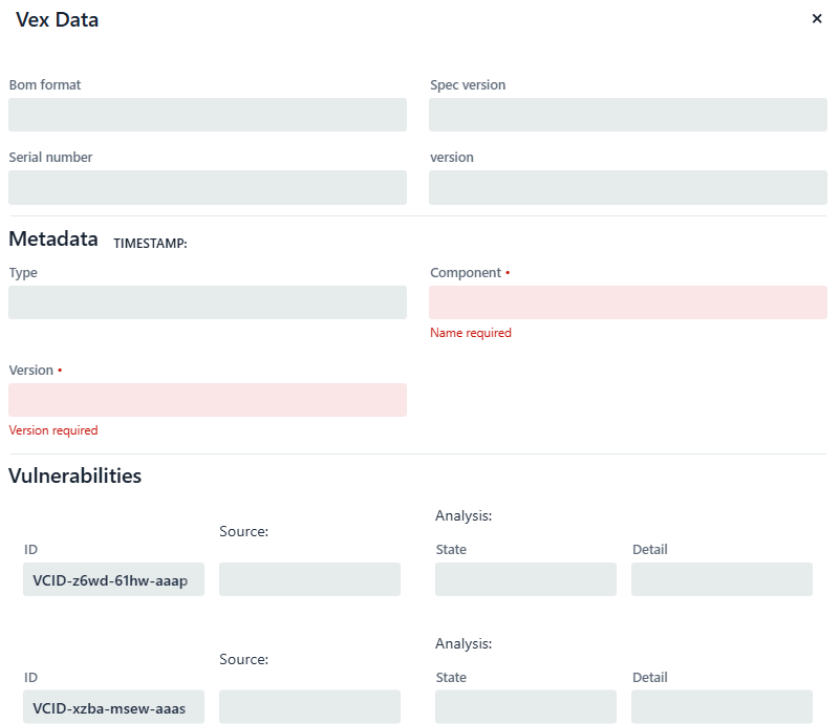


Figure A1-6: Already in the overview, those SW components that have a vulnerability can be filtered

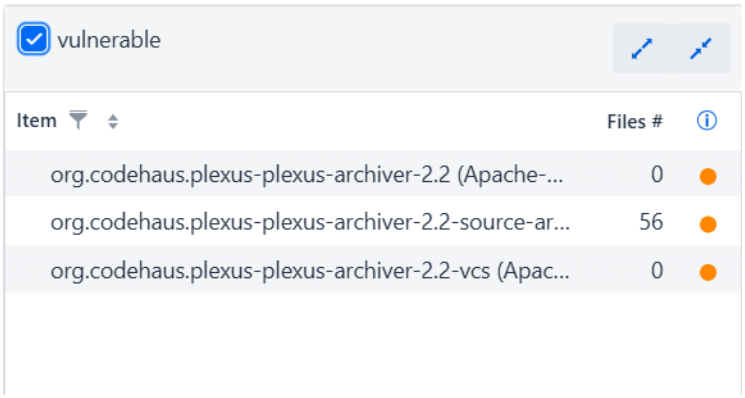


Figure A1-7: The sources of a SW product are clearly displayed, filterable and searchable

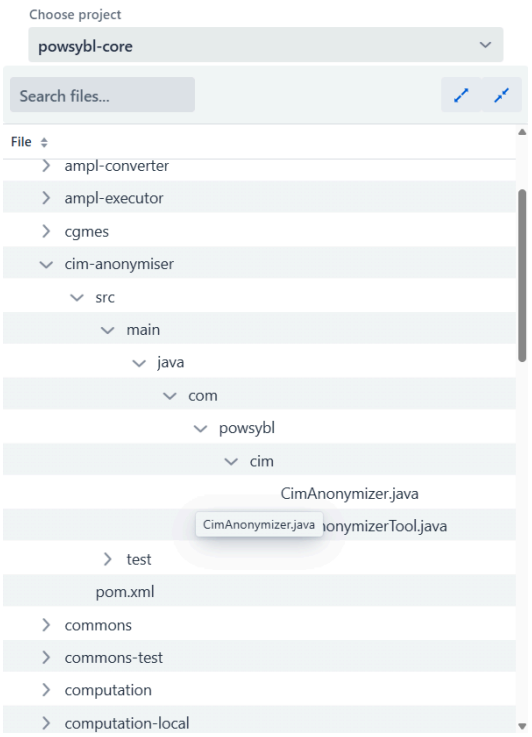
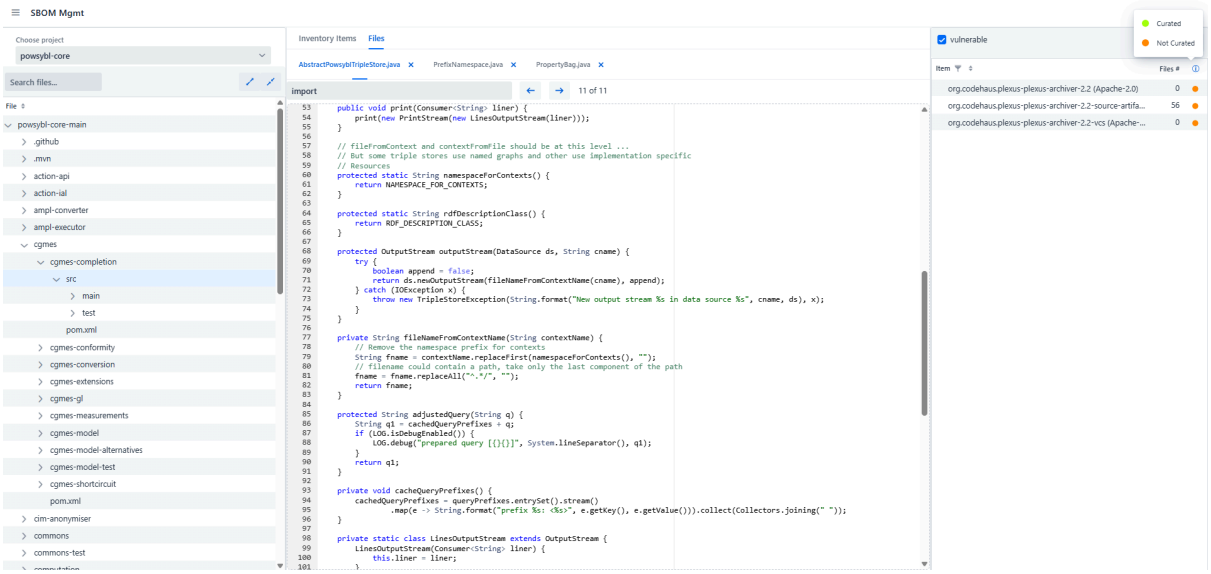


Figure A1-8: If the source code is considered for the analysis of the vulnerability, this is possible



9 Annex 2 - Octet-curator Reference Frameworks and Sources

The following table shows the composition of tools relied upon by the Bitsea OCCTET curator.

Tool	version	URL	Description
Jmix framework	2.6	https://www.jmix.io/	Java full-stack framework used to build the frontend UI and entity-based application logic.
Spring AI	1.0.0-M6	https://docs.spring.io/spring-ai/docs/	Spring module used to interface with AI models (via Ollama) for tasks like filtering, license analysis, and summarization.
Spring Boot	?1.0.0-M6	https://spring.io/projects/spring-boot	Backend microservices and frontend integration Powers all microservices; each microservice is a small Spring Boot app performing a single task.
Ollama	latest	https://ollama.com/	

Tool	version	URL	Description
			Local framework for running LLMs like LLaMA or Mistral, integrated into the AI microservice.
Postgresql	16.0	https://www.postgresql.org/	Main database storing all entities, inventory items, metadata, audit results, and SBOM data
Pg_vector	Extension	https://github.com/pgvector/pgvector	Enables hybrid retrieval and document embedding storage in PostgreSQL.
Pg_trgm	Extension	https://www.postgresql.org/docs/current/pgtrgm.html	Supports fuzzy searching and filtering in metadata-heavy fields.
NATS / Jetstream NATS	2.22.0	https://docs.nats.io/nats-concepts/jetstream	Provides persistent messaging and work-queue-style task distribution across microservices.
Gradle	8.12.1	https://gradle.org/	Build automation tool used for compiling, testing, and packaging each microservice and the frontend. Supports creating runnable JARs or Docker images.
Maven	3.9.11	https://maven.apache.org/index.html	Maven is a build automation tool used primarily for Java projects. It provides

Tool	version	URL	Description
			a standardized way to build, package, and deploy projects. It simplifies the build process by using a Project Object Model (POM) file, typically `pom.xml`, which describes the project's dependencies, build lifecycle, and other configuration details.
Java	21.0.7	https://openjdk.org/	Java is a widely-used, class-based, object-oriented programming language designed to have as few implementation dependencies as possible.

10 ACRONYMS AND ABBREVIATION

CRA — Cyber Resilience Act
ENISA — European Union Agency for Cybersecurity
EU — European Union
GDPR — General Data Protection Regulation
FOSS — Free and Open-Source Software
AI — Artificial Intelligence
IPR — Intellectual Property Rights
KPI — Key Performance Indicator
PII — Personally Identifiable Information
RBAC — Role-Based Access Control
SBOM — Software Bill of Materials
SAST — Static Application Security Testing
DAST — Dynamic Application Security Testing
MFA — Multi-Factor Authentication
SOC — Security Operations Centre
IR — Incident Response
API — Application Programming Interface
TLS — Transport Layer Security
ISO — International Organization for Standardization
IEC — International Electrotechnical Commission
ETSI — European Telecommunications Standards Institute
SUS — System Usability Scale
TRL — Technology Readiness Level
WP — Work Package
DoA — Description of Action

11 BIBLIOGRAPHY

1. **European Commission**, The Cyber Resilience Act — Questions & Answers, European Commission, 2024.
2. **European Commission**, DIGITAL Europe Programme – Model Grant Agreement, European Commission, 2024.
3. **OCCTET Project Consortium**, Description of Action (DoA), Grant Agreement No. 101190474, 2024.
4. **OCCTET Project Consortium**, CRA SME requirements and self-assessment checklists (D1.2), 2025.