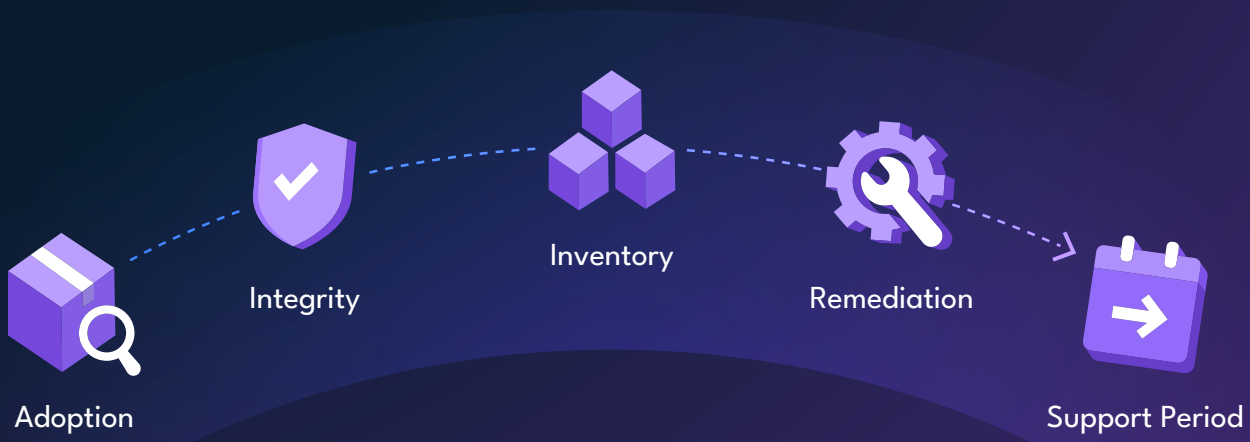
Guide

TuxCare

Open Source Compliance Across the Package Lifecycle

What the EU CRA, FedRAMP, PCI DSS, NIS2, and DORA require of the open source in your products, and the evidence that proves it

For the engineering, security, and compliance teams that ship open source



REGULATIONS COVERED

EU CRA | FedRAMP | PCI DSS | NIS2 | DORA

TuxCare, November 2026

Contents

What this guide covers

03 Executive Summary
The four demands the five regimes share, and the dates that matter.

04 Why does every regulator now care about the open source inside your product?
The four demands the five regulations share and the deadlines that have already passed.

08 How do you turn a regulation clause into a requirement for a package?
A three-column method for translating any clause into an engineering obligation and an audit artifact, and the ten artifacts that serve as evidence.

12 What must you do before you adopt an open source package?
The due-diligence decision to write down before integration, the screening each new version goes through, and what an intake record contains.



16 How do you prove the package you run is the one the maintainer published?
What the regulations require for integrity, and why signatures and provenance are the evidence, although no regulation except FedRAMP's High baseline names them.



19 What must your component inventory contain, and how deep must it go?
Why every regime wants an SBOM, why they disagree about its depth, and how VEX helps you fix what affects you inside the deadlines.



23 How fast must you fix a vulnerability, and when must you report it?
Patch deadlines and reporting deadlines, measured differently, and both falling on whoever ships or runs the software.



28 What must you do when a project stops maintaining a component you use?
A practical model for unsupported components, and the ten-year misreading to avoid.



31 Which regulations apply to you, and which of their clauses reach your open source components?
One section per regulation: scope, dates, penalties and a clause table by lifecycle stage.
8.1 EU CRA; 8.2 FedRAMP; 8.3 PCI DSS 4.0.1; 8.4 NIS2; 8.5 DORA; 8.6 SOC 2, ISO 27001 and CIS Controls.

43 What evidence must you be able to show an assessor?
The ten artifacts in one table, with owners and timing, and a one-page self-assessment.

46 Where SecureChain and ELS fit in the lifecycle
What a SecureChain or ELS subscription supplies at each stage, TuxCare's status as your supplier, and what it does not cover.

49 Appendix A. The regulatory timeline, 2024 to 2027

50 Appendix B. Glossary

51 Appendix C. Primary texts

EXECUTIVE SUMMARY

The rules already apply to the open source you ship.

Five regimes now reach into the dependency tree of software sold or run in regulated markets: the EU Cyber Resilience Act, the US FedRAMP program, the PCI Data Security Standard, the NIS2 Directive and the EU's Digital Operational Resilience Act. Most of their requirements are technology-neutral, and they reach open source whether or not they name it.

Read side by side, they converge on four demands. You have to know what you ship, vet what you adopt, fix what breaks inside a defined response window, and keep fixing it for as long as you said you would. Compliance programs, and the guidance written for them, usually concentrate on the inventory and the fix. Vetting is where the CRA and DORA are most explicit, and continued support is where long-running gaps tend to surface.

This guide follows a package through its life, from the day someone decides to use it to the day its project stops maintaining it, and asks at each stage what the regulations require, what that means in practice, and what evidence closes the requirement. It is written for three readers at once: the developer who adds the package, the security engineer who has to defend it, and the compliance specialist who has to prove it.

Each chapter stands alone. The regulation names appear in the text; the article and clause numbers wait at the end of each chapter and in Chapter 8, for whoever needs them.

On 11 September 2026, the CRA's reporting obligations took effect, including for products already on the EU market. On 7 December 2026, FedRAMP's new vulnerability rules become mandatory for every authorized cloud service. On 11 December 2027, the CRA applies in full. The first of those has already passed. The other two are closer than most roadmaps assume.

ALREADY IN FORCE

11 Sep 2026

CRA reporting obligations apply, including to products already on the market

COMING NEXT

7 Dec 2026

FedRAMP 2026 vulnerability rules mandatory for every authorized service

COMING NEXT

11 Dec 2027

CRA applies in full: essential requirements, conformity assessment, CE marking

Legal note. This publication is for informational purposes only and does not constitute legal advice. The regulations it describes are complex, several are still being completed through implementing acts, delegated acts, harmonized standards and official guidance, and their application to a specific product or organization depends on facts this guide cannot know. Verify the current status of any regulatory reference before acting on it, and consult qualified counsel for decisions that depend on it.

CHAPTER 1

Why does every regulator now care about the open source inside your product?

THE SHORT VERSION

Five regimes, written for five different purposes, now make the same four demands of the open source inside your products. Their requirements are mostly technology-neutral, and they reach open source whether or not they name it. If you sell software into the EU, sell cloud services to the US federal government, handle card payments, run essential infrastructure in Europe, or serve financial institutions there, at least one of them already applies to you.

A few years ago the question "is our open source compliant?" meant licensing. Did the terms of the components you used allow what you were doing with them? Legal usually answered it, often late in the release cycle.

That question has not gone away, but it is no longer the one regulators are asking.

The question now is about security, and it is asked of the product rather than the organization. What is inside it? Where did each piece come from? Who is fixing it, how quickly, and for how long? The regulations that ask these questions were not written with open source in mind. They were written about "components," "third-party software," "ICT assets" and "products with digital elements." ICT, a term the EU texts use throughout, means information and communication technology, which in practice means any IT system. Open source is simply most of what those words describe.

The regulations reach open source whether or not they name it

The EU Cyber Resilience Act makes a manufacturer responsible for every component in a product it sells into the EU, and it says outright that this includes free and open source components that were never supplied commercially. The regulation also requires a manufacturer that finds a vulnerability in an integrated component, open source included, to report it to whoever maintains that component and to share any fix it develops. The European Commission's guidance of July 2026 confirms that these duties apply to any open source component a manufacturer integrates.

DORA's technical standard for financial entities is the most direct of all. It requires banks and insurers to track the third-party libraries, "including open-source libraries," used by ICT services that support critical or important

functions, and it requires source code coming from open source projects to be analyzed and tested, where feasible, before it reaches production. Both requirements are written into the binding technical standard's own text.

FedRAMP's control requirements do not mention open source, and they do not need to, because they apply to every system component. The supplementary guidance to one of them does mention it: the control that governs what happens when a component is no longer supported names "open-source software value-added vendors" as one acceptable source of continued support.

PCI DSS asks for an inventory of the third-party software inside your payment software so that you can patch it.

NIS2 asks essential and important entities to weigh the vulnerabilities and the development practices of each supplier. Its implementing regulation asks the digital entities it covers for a complete and current asset inventory, for which the regulation's own recital and ENISA's guidance list each asset's last patch date and end of life among the fields to consider.

Taken together, the regulations cover the whole life of a software component: the decision to adopt it, the way it reaches you, what you know about it, how fast you fix it, and what you do after its maintainers stop.

The four demands the five regulations share

Most writing about compliance and open source reduces the regulations to three demands: keep an inventory, remediate within a timeframe, and have a documented process. That list is accurate, but it leaves out the stage where the regulations have become most explicit, and where most organizations have no evidence at all.



Know what you ship

Every regime wants an inventory of the components in your product or environment, increasingly in machine-readable form.

CHAPTER 5



Vet what you adopt

A documented basis for trusting a component before it is integrated, and proof that what you run is what the maintainer published.

CHAPTERS 3 AND 4



Fix inside a defined window

Every regime imposes a remediation obligation; only PCI DSS and FedRAMP set numbers of days.

CHAPTER 6



Keep fixing it

Your obligation does not end when a maintainer's does.

CHAPTER 7

The CRA, DORA and NIS2 each require a documented basis for trusting a component before it is integrated, and the CRA, FedRAMP and DORA each expect you to know that the component you run is the one its maintainer published; PCI DSS expects it of the scripts on a payment page. On remediation, the CRA, NIS2 and DORA use standards such as "without delay" or require you to set and meet your own deadlines. On support, the CRA sets a minimum support period, FedRAMP gives you two acceptable states for an unsupported component, and PCI asks for a management-approved plan for every technology approaching end of life.

A compliance program that covers only the first and third demands is half built.

The deadlines that have passed and the two that are next

Regulation arrives in stages, and the stage that matters most is rarely the one with the headline date. The CRA's headline date is 11 December 2027, when every product with digital elements sold in the EU must meet its essential requirements and carry a CE mark. But its reporting obligations took effect on 11 September 2026, and they apply to products already on the market. If one of your products contains an actively exploited vulnerability today, you have 24 hours from learning about it to file an early warning. That clock runs whether the vulnerable component is your own code or an open source package four levels down your dependency tree.

The CRA is the newest of the three EU regimes in this guide. NIS2 has been national law since October 2024 and DORA has applied to financial entities since January 2025, so their supply chain and asset requirements are already being supervised.

FedRAMP's new vulnerability rules became optional in July 2026 and will become mandatory on 7 December 2026. They replace the fixed 30-, 90- and 180-day windows that most compliance teams still have written into their procedures with a risk-based model that expects a fix or a mitigation for the most dangerous findings in two days at the Moderate baseline and in 12 hours at the High baseline. Services that have not adopted the rules by 7 March 2027 risk losing their authorization.

PCI DSS version 4 finished arriving on 31 March 2025, when all of its future-dated requirements became mandatory, including the inventory of third-party software components.

Most of these obligations are already in force, and the rest have fixed dates. The practical question is whether your evidence can show compliance today, for the versions currently in production. The chapters that follow each take one moment in a package's life and ask what the regulations require at that moment.

AT A GLANCE

Who demands what, stage by stage

What each regulation requires at each of the five lifecycle stages. Chapters 3 to 7 take one stage each.

STAGE	EU CRA	FedRAMP	PCI DSS	NIS2	DORA
Adoption Ch. 3	● Check every third-party component before it goes in	● Document how components enter the system and block counterfeits	● Add each new component to the payment software inventory	◑ Assess each supplier's security record and practices	● Analyze and test open source code before production, where feasible
Integrity Ch. 4	● Protect the product from tampering and ship updates securely	◑ Detect unauthorized changes and verify signatures	◑ Authorize and monitor scripts on payment pages	◑ Check that patches come from trusted sources and are intact	● Prevent tampering during development and deployment
Inventory Ch. 5	● Keep an SBOM; direct dependencies are the legal minimum	● Keep an inventory of every system component	● List the third-party components inside payment software	● Keep a complete, current asset inventory	● Inventory IT assets and track their open source libraries
Remediation Ch. 6	● Fix “without delay”	● Fix the worst findings in 12 hours (High) or 2 days (Moderate)	● Patch critical flaws within a month; set your own schedule for the rest	● Fix critical vulnerabilities “without undue delay”	● Scan weekly and meet the patch deadlines you set
Support period Ch. 7	● Support the product for at least the minimum period	● Replace each unsupported component or get continued support	● Keep an approved plan for technology nearing end of life	◑ Record end-of-life dates in the asset inventory, as guidance suggests	◑ Record when each supplier's support ends

● Full requirement ◑ Partial or narrower scope High, Moderate: FedRAMP baselines

Where this comes from: Regulation (EU) 2024/2847 (Cyber Resilience Act), Articles 13(5), 13(6), 14 and 71; European Commission CRA guidance C(2026) 5252 of 27 July 2026, paragraphs 88 and 170 to 173. Commission Delegated Regulation (EU) 2024/1774 (DORA RTS), Articles 10(2) and 16(8). NIST SP 800-53 Rev. 5, control SA-22. PCI DSS v4.0.1, Requirement 6.3.2. Directive (EU) 2022/2555 (NIS2), Article 21(2) and (3); Implementing Regulation (EU) 2024/2690, Annex point 12.4; ENISA Technical Implementation Guidance, June 2025, section 12.4. FedRAMP 2026 Consolidated Rules, Vulnerability Detection and Response.

CHAPTER 2

How do you turn a regulation clause into a requirement for a package?

THE SHORT VERSION

Every clause that touches software can be read in three columns: what the text says, what it means for a package, and what evidence closes it. The evidence comes from a short list of ten artifacts. Learn the ten and you can read any of the five regimes without a lawyer at your elbow.

Regulations are written about categories: "components," "third-party software," "ICT assets," "products with digital elements." Nobody in your engineering organization has ever adopted a product with digital elements. They have added a package, pinned a version and moved on.

The translation between those two vocabularies is the whole job of compliance for open source, and it is simpler than it looks.

The three-column method for reading a clause

Take any clause and write it out in three columns. The first column is the clause as written, quoted rather than paraphrased, because regulators choose their words with care and the difference between "at least" and "all," or between "without delay" and "within 30 days," is where audits are won and lost.

The second column is what the clause means for a package. This is the step the regulation never does for you, because the regulation does not know you use packages. Ask: at which moment in a package's life does this clause bite, and what would I have to do to the package, or know about it, to satisfy the words in column one?

The third column is the artifact that proves you did it: the record showing you did, for this package, on this date, as opposed to the policy saying you will.

PCI DSS never uses the term SBOM, but an SBOM is how assessors expect its component inventory requirement to be met: a complete, versioned component list, ready to match against new vulnerability data.

1 The clause as written	2 What it means for a package	3 The artifact that proves it
Quoted, not paraphrased. "At least" versus "all" is where audits are won and lost.	The step the regulation never does for you: when does the clause bite, and what must you do or know?	The record that you did it, for this package, on this date. Not the policy that says you will.
Example. PCI DSS 6.3.2: an inventory of third-party software components, maintained "to facilitate vulnerability and patch management."	**Example.** Every package in a payment application, including transitive ones, listed so you can tell quickly whether a new vulnerability affects you.	**Example.** An SBOM generated for each release, kept for each version in production.

The three-column method, applied to PCI DSS. Chapter 8 does this for every clause of the five regimes that reaches an open source component.

The ten artifacts auditors accept as evidence

The third column keeps producing the same answers, and the same ten artifacts cover nearly every clause in the five regimes. If these terms are new to you, read this section before the rest of the guide.



1. Component source policy · Ch. 3

Where packages may come from and which checks they must pass. Required outright by NIS2 and FedRAMP.



2. Component intake record · Ch. 3

Why a component was allowed in. Answers "show me how this component was vetted."



3. Screening result · Ch. 3

Per-version check for malware, look-alike names and install-time behavior.



4. SBOM · Ch. 5

Machine-readable component list. Depth matters: top-level or full tree.



5. VEX statement · Ch. 5

Whether a listed vulnerability affects your product, and why.



6. Provenance attestation · Ch. 4

Signed record of how an artifact was built: source, system, conditions.



7. Verified signature · Ch. 4

Shows a package was not altered since it was signed.



8. Remediation log · Ch. 6

Dated record of each vulnerability, from detection to deployed fix.



9. Support agreement with SLA · Ch. 7

The contract that turns "we will patch this" into evidence.



10. Risk acceptance record · Ch. 7

A written, time-bound decision to live with a known risk.

The ten artifacts at a glance. Full definitions follow.

A component source policy is a written statement of where packages may be obtained from and which checks they must pass before they enter your environment. It makes the adoption decision once, for everyone, and it is the one artifact in this list that two regimes require outright: NIS2's implementing regulation asks covered entities for a supply chain security policy, and FedRAMP's baselines ask for documented supply chain processes and an anti-counterfeit policy. Chapter 3 covers it.

A component intake record is the dated record of why a component was allowed in: where it came from, who maintains it, what license it carries, whether the project publishes advisories, which vulnerabilities were open at adoption and what you decided about them. Assessors know it as a third-party component assessment. It is the answer to "show me how this component was vetted."

A screening result is the record that a given package version was checked, before it enters your environment, for known malware, for names that imitate popular packages and for what the code would do once installed and run. It is produced per version, usually by the registry or proxy you pull from, and it is the evidence behind the clauses on malicious software.

A software bill of materials, or SBOM, is a machine-readable list of the components in a piece of software, with their versions and, in the better ones, who produced each component and what license it carries. The two common formats are CycloneDX and SPDX. The choice that matters is depth: a top-level SBOM lists the packages you chose, while a full SBOM also lists the packages those packages pulled in, which is where most vulnerabilities live. A usable SBOM also records whether each component's upstream still maintains it and when its support ends. Chapter 5 covers depth and Chapter 7 what to do when that support ends.

A VEX statement is a short machine-readable note saying that a particular vulnerability in a listed component does, or does not, affect your product, and why. Without VEX, a scan of an SBOM generates hundreds of findings, most of which do not apply to you. With VEX, each finding has a status an auditor can read. No regulation requires VEX, but nearly every team that uses SBOMs at scale ends up needing it.

A provenance attestation is a signed record of how a software artifact was built: from which source, by which system, under what conditions. The common vocabulary for how strong a provenance claim is comes from the SLSA framework, which defines levels. A provenance attestation answers the question "did this package really come from where it says it did?"

A verified signature on a package shows that it has not been altered since it was signed by the holder of a particular key. It does not show that the key holder is the legitimate maintainer, or that the signed code is safe. Signatures and provenance are the evidence behind every clause about integrity and secure distribution. Chapter 4 covers them.

A remediation log is a dated record of each vulnerability: when it was detected, how it was triaged, when a fix was available, when the fix was deployed. Every clock in Chapter 6 is measured against it.

A support agreement with a service level is the contract that turns "we will patch this" into evidence. It matters most after a project's own maintainers have stopped, which is Chapter 7.

A risk acceptance record is the written decision to live with a known risk for a stated period, with an owner and an end date. It is the artifact that distinguishes a managed exception from neglect.

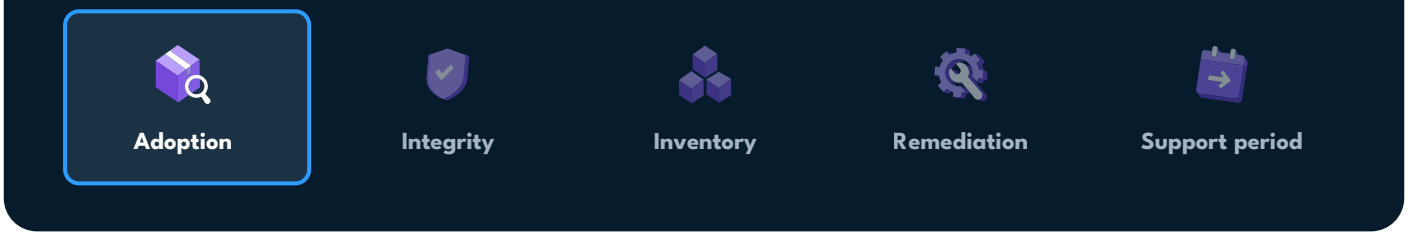
One caution about all ten. None of the five regimes requires VEX or names SLSA, the CRA never says "signed," and PCI DSS never says SBOM. The regulations set outcomes, and these artifacts are how the industry demonstrates them, so present them to an assessor as your evidence for meeting a requirement rather than as something the regulation mandates. There are three exceptions: the component source policy, which NIS2's implementing regulation and FedRAMP's baselines require outright; the SBOM, which the CRA names; and signature verification, which is required by FedRAMP's High baseline.

Keep the quoted requirement and the artifact that answers it side by side.

Where this comes from. PCI DSS v4.0.1, Requirements 5 and 6.3.2. Regulation (EU) 2024/2847, Article 13(5) and Annex I Part II points (1), (7) and (8). Implementing Regulation (EU) 2024/2690, Annex points 5.1 and 6.9. NIST SP 800-53 Rev. 5, controls SR-3, SR-11 and SI-3 (all baselines), SR-4 (Provenance, not allocated to any baseline) and CM-14 (Signed Components, FedRAMP High baseline only). CISA, 2026 Minimum Elements for a Software Bill of Materials, version 2.1, 29 July 2026. SLSA specification, slsa.dev. CISA VEX documents.

CHAPTER 3

What must you do before you adopt an open source package?



THE SHORT VERSION

The decision to adopt an open source component is now a regulated decision, and all five regimes reach it. The CRA requires due diligence on every third-party component before it goes into a product, and DORA requires open source code to be analyzed and tested, where feasible, before it reaches production. NIS2 asks you to weigh each supplier's security practices, FedRAMP expects a documented acquisition process, and PCI DSS expects an inventory that starts when the component is adopted. If that decision lives only in a developer's memory, you have no evidence that it was made. And it is not made once: every new version is new third-party code entering your product.

Adding a package takes one command and, in most organizations, leaves no record. The developer chooses, the build picks it up, and the first time anyone outside the team learns the package exists is when a scanner flags it, months later, several layers deep.

That working practice is now a compliance gap, and it is the gap the regulations have become most explicit about.

The CRA due-diligence requirement for third-party components

The CRA states it directly: a manufacturer must exercise due diligence when integrating components sourced from third parties, so that those components do not compromise the security of the product. The obligation extends to free and open source components that were never made available commercially. The European Commission's guidance of July 2026 confirms the scope: the duty applies to any open source component a manufacturer integrates, and so do the regulation's separate duties to report

vulnerabilities found in it to the project and to share any fixes.

"Due diligence" is a legal phrase. In practice it means a documented reason to trust the component, proportionate to the risk it carries. For a utility that formats dates, that reason can be short. For a component that handles authentication or parses untrusted input, it cannot.

The CRA does not prescribe a method; the Commission's guidance describes risk-based verification through specifications, documentation or testing. The one answer that will not pass is no verification at all.

The DORA requirement to test open source code before production

DORA's technical standard for financial entities names open source at this stage. Where feasible, source code from open source projects must be analyzed and tested, with static and dynamic review, before it is deployed to production, and the use of open source libraries in systems supporting critical or important functions must be tracked. Tracking has to begin at adoption. For a bank or an insurer in the EU this is written law, and for anyone who builds software for one it arrives as a contractual requirement.

How NIS2, FedRAMP and PCI DSS reach the adoption decision

The other three regimes reach the adoption decision through supplier, process and inventory requirements rather than through a clause written about components, which is why this guide treats the CRA and DORA first. The obligation is no lighter; it is simply stated less directly.

NIS2 asks the same question from the supplier side. When essential and important entities decide their supply chain measures, they must take into account the vulnerabilities specific to each supplier and the supplier's secure development practices. An open source project is not a supplier under contract, but the same question applies to every project you depend on: how does this code get maintained, and how does it get fixed? For the covered digital entities, a second requirement lands at the same point: they must prevent or detect the use of unauthorized software, a requirement that bites at the moment a package is installed.

FedRAMP's baselines include a family of supply chain controls and an acquisition process control. Together they expect a cloud service provider to have a documented process for how components enter the system, in addition to how they are scanned once inside. Two further controls bear on what comes in: malicious code protection, and component authenticity, which requires a policy and procedures to detect and prevent counterfeit components from entering the system.

PCI DSS asks for the inventory of third-party components inside your payment software. An inventory reconstructed at audit time is a scan result. The requirement is met by a record that starts when the component is adopted and is updated when it changes. PCI DSS also requires protection of all systems against malicious software, and that covers the components in the inventory as much as the hosts they run on.

Screening a package before it enters the environment

DORA is the only regulation of the five that says the code itself must be analyzed and tested before it is deployed. FedRAMP, PCI DSS and NIS2 require protection against malicious software without saying that the check has to happen when a package comes in, and the CRA leaves the method of due diligence open. In each case the moment a package comes in is the place for this check, because a package can come from a well-run project and still be malicious.

Registry accounts can be taken over and used to publish poisoned versions of trusted packages. Look-alike packages can be published under names one character away from the real ones to catch developers who mistype an install command. Maintainers can hand projects to strangers who turn them malicious. In every one of those cases, a package reaching your build is exactly the one its publisher released, and nothing in its record warns you.

Screen each package, and each new version of it, before it enters your environment, and keep the result of each screening as the record that the package passed.

SCREEN EVERY VERSION FOR

- **Known malware**
- **Names that imitate popular packages**
- **What the code does once installed and run**

What a component intake record contains

The artifact for this stage is a component intake record. It is less than a security review and more than a line in a lock file. For each component you adopt, it records the fields below.

Component intake record	
Source	Where it came from
Maintainers	Who maintains it, how actively
License	Which license it carries
Advisories	Publishes advisories; has a reporting channel
Open CVEs	Known at adoption, and the decision taken
Integrity	Signature or provenance available and verified
Screening	Result of the screening it passed
Decision	Risk judgment, owner, date

Most of those fields can be filled automatically from the package's own metadata and public vulnerability data. The ones that cannot, the risk judgment and the decision, are the ones an assessor actually reads.

Pair the record with a short written policy on where components may be obtained from. A curated internal source with known screening rules satisfies several clauses at once, because it makes the due-diligence decision once, for everyone, instead of leaving it to each developer at each install.

Decide where packages may come from and write down why each one was allowed in at the moment it is adopted and each time a new version arrives; every later stage is easier when that record exists.

THE AUDIT QUESTION FOR THIS STAGE

"Show me how this component was vetted before it was integrated."

If the answer is a developer's recollection, you do not have evidence. If the answer is an intake record, a source policy and a screening result dated before the first production deployment, you do.

EVIDENCE AN ASSESSOR EXPECTS

- **Intake record**
- **Source policy**
- **Screening result, dated before first deployment**

Where this comes from. Regulation (EU) 2024/2847, Article 13(5) and 13(6); European Commission CRA guidance C(2026) 5252, 27 July 2026, paragraphs 88 and 170 to 173. Commission Delegated Regulation (EU) 2024/1774, Article 16(2), (3) and (8), and Article 10(2). Directive (EU) 2022/2555, Article 21(2)(d), (e) and 21(3); Implementing Regulation (EU) 2024/2690, Annex points 5.1, 5.2 and 6.9. NIST SP 800-53 Rev. 5, controls SR-3, SR-5, SR-11, SA-4 and SI-3. NIST SP 800-218 (SSDF), practice PW.4. PCI DSS v4.0.1, Requirements 5 and 6.3.2.

CHAPTER 4

How do you prove the package you run is the one the maintainer published?



THE SHORT VERSION

The CRA, FedRAMP, PCI DSS and DORA each require you to protect the integrity of the software you ship or run. Apart from FedRAMP's High baseline, none of them says what evidence satisfies the requirement. Signatures and provenance are the evidence an assessor expects.

The package you installed may not be the package you chose. A build system or a distribution server can be compromised, and the package it produces or serves will no longer match the source code in the project's own repository. The difference exists only in the file you installed, while everything about it says it is the one you asked for. Integrity evidence is how you show that it is: a signature, which proves the artifact was not altered after the maintainer released it, and provenance, which proves it was built from the source the maintainer published.

What the five regulations require for software integrity

The CRA's essential requirements say a product must protect the integrity of its programs and configuration against any manipulation not authorized by the user, and must have mechanisms to distribute updates securely. Those two requirements cover the package you install and the update you push.

FedRAMP's baselines include a control on software and firmware integrity that requires integrity verification tools to detect unauthorized changes. It applies at the Moderate and High baselines. The control on provenance exists in the NIST catalog but is in no FedRAMP baseline. The control on signed components is in the High baseline only, so verifying signatures before installation is a requirement at the High baseline and good practice below it.

PCI DSS applies the same logic where payment data is most exposed. Scripts that run in the customer's browser on a payment page must be authorized, their integrity assured and an inventory of them kept, and a change- and tamper-detection mechanism must alert you to unauthorized modification.

DORA's technical standard requires measures against intentional manipulation of systems during development, maintenance and deployment, and controlled change management for software components.

NIS2 comes closest in its patch management rule: for covered digital entities, security patches must come from trusted sources and be checked for integrity.

Each of them requires the same outcome: the code that runs must be the code that was meant to run.

Why signatures and provenance are evidence rather than requirements

None of these texts says "attestation" or "SLSA." The CRA says "securely distribute." FedRAMP says "integrity verification tools," and only its High baseline goes as far as "digitally signed." DORA says "mitigate the risk of unintentional alteration or intentional manipulation."

Signatures, provenance attestations and verified builds are how the industry demonstrates those outcomes. When an assessor asks how you know the artifact in production was not altered on its way to you, a verified signature and a provenance record are the answer they expect. Claim that you meet the outcome and can demonstrate it with the artifact.

THREE PRACTICES THAT PRODUCE INTEGRITY EVIDENCE

1



Verify signatures in the pipeline, and log the verification

A signature that is present but never checked proves nothing.

2



Collect provenance attestations

Where the publisher offers them, and record the SLSA level the attestation actually supports rather than the level you would like to claim.

3



Pull from a private registry or proxy

One that verifies signatures and collects provenance on your behalf, so the policy is applied once rather than relying on each developer at each install. Its logs become the evidence that every component passed the checks.

Each of these practices produces its own evidence.

THE AUDIT QUESTION FOR THIS STAGE

"How do you know this artifact was not tampered with between the public registry and your build?"

The answer the assessor is looking for is a verification log and a provenance attestation for the exact artifact in production.

EVIDENCE AN ASSESSOR EXPECTS

- **Signature verification log**
- **Provenance attestation for the exact artifact**

Where this comes from. Regulation (EU) 2024/2847, Annex I Part I point 2(f), Annex I Part II point (7). NIST SP 800-53 Rev. 5, controls SI-7 (Moderate and High baselines); CM-14 (High baseline); SR-4 (not allocated to any baseline). PCI DSS v4.0.1, Requirements 6.4.3 and 11.6.1. Commission Delegated Regulation (EU) 2024/1774, Article 16(1)(c); Regulation (EU) 2022/2554, Article 9(4)(e). Implementing Regulation (EU) 2024/2690, Annex point 6.6.1(c).

CHAPTER 5

What must your component inventory contain, and how deep must it go?


Adoption


Integrity


Inventory


Remediation


Support period

THE SHORT VERSION

Every regime expects an inventory of the components in your software. They disagree about how deep it must go. The CRA's legal floor is your direct dependencies. CISA's 2026 Minimum Elements for an SBOM (version 2.1), written with seventeen partner agencies, expect the full dependency graph, with hashes and licenses. Build to the deeper standard: an SBOM that meets it also meets the CRA floor, and only the deeper one can answer the question every incident clock in Chapter 6 starts with, which is whether you are affected.

Of the four demands the five regulations share, the inventory is the one on which all five agree. The CRA requires a software bill of materials. PCI DSS requires an inventory of third-party components. FedRAMP requires a component inventory. DORA requires inventories of ICT assets and tracking of open source libraries. NIS2's implementing rules require an asset inventory, and the regulation's recital and ENISA's guidance suggest patch and end-of-life dates for it.

The CRA minimum depth and the CISA reference standard

The CRA sets the floor: the SBOM must be in a commonly used, machine-readable format and must cover "at the very least the top-level dependencies" of the product. Top-level means the packages you chose directly. The regulation permits you to stop there. It also says the Commission may later specify the format and elements of the SBOM in an implementing act; as of this writing it has not.

The reference standard is moving the other way. In July 2026 the US Cybersecurity and Infrastructure Security Agency published version

2.1 of its Minimum Elements for a Software Bill of Materials, co-authored with seventeen partner agencies including Germany's BSI, France's ANSSI and the Dutch NCSC. It expanded the minimum expected data fields from seven to seventeen, adding a cryptographic hash for each component, its license, and the context in which the SBOM was generated. It replaced the old notion of "depth" with "coverage," and coverage means the complete dependency graph rather than the first layer alone. CycloneDX and SPDX remain the accepted formats.

7 → 17 minimum data fields in CISA's SBOM elements, version 2.1 (July 2026)

So an SBOM that meets only the CRA's floor fails the reference standard, and an SBOM that meets the reference standard satisfies the CRA's floor by inclusion. There is no conflict to resolve; one choice satisfies both.

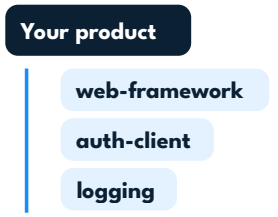
Why a top-level SBOM cannot tell you whether you are affected

Consider what the inventory is for. The CRA's reporting obligation gives you 24 hours from learning of an actively exploited vulnerability to file an early warning. NIS2 gives essential and important entities the same 24 hours for a significant incident. FedRAMP's 2026 rules can require remediation within 12 hours at the High baseline. Every one of those clocks starts with the same question: are we affected?

A top-level inventory cannot answer it. The vulnerable component is, more often than not, several layers down in something you did not choose and may never have heard of. A full inventory answers the question in minutes, by machine. A top-level one answers it in days of manual work.

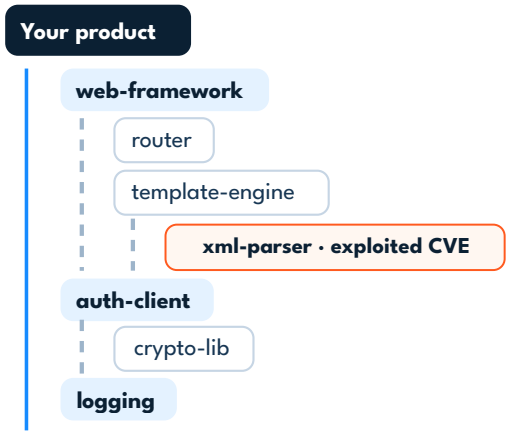
The CRA permits a top-level SBOM, and the same regulation's 24-hour reporting clock is very hard to meet with one.

TOP-LEVEL SBOM · CRA FLOOR



Three packages listed. The exploited parser is invisible, and the 24-hour clock is already running.

FULL SBOM · CISA 2.1 COVERAGE



The vulnerable parser three layers down is listed. “Are we affected?” takes minutes.

Illustrative example. Package names are generic.

The lifecycle fields DORA and NIS2 add to the inventory

DORA's technical standard also requires asset records to carry the end dates of the supplier's support, and NIS2's implementing rules require covered entities to keep a complete and current asset inventory, for which the regulation's recital and ENISA's guidance list a last-patch date and an end-of-life date among the fields to consider. Those fields come from combining the SBOM with lifecycle data about each component, meaning when it was last patched and when its project stopped or will stop maintaining it. No SBOM generator produces that data on its own, and Chapter 7 depends on having it.

Why an SBOM needs VEX statements

A full SBOM fed into a vulnerability scanner produces a long list of findings. Most entries will be vulnerabilities in code paths your product never calls, or in components present only at build time, or already fixed in your deployed version. Without a way to say so, every entry is an open finding. The penalties in these regimes attach to vulnerabilities that affect you and go unfixed; scanner entries that do not apply to you carry no penalty, but they consume the same attention until someone shows they do not apply, and a team that cannot do that quickly ends up either chasing hundreds of non-issues or distrusting the list.

VEX solves this. A VEX statement records, per vulnerability and per product, whether you are affected and, if not, why not. It turns a scanner's output into a defensible position. No regulation requires it, but teams that run full SBOMs at scale end up building something like it.

What to generate and retain for each release

Generate an SBOM for every release in CycloneDX or SPDX, from the build itself rather than from source, so that it reflects what actually shipped. Keep one per version for as long as that version is in production, plus the period any regime requires you to retain documentation. Attach VEX statements as findings are triaged. Add each component's lifecycle data so the inventory can answer "when was this last patched" and "when does this stop being maintained" as well as "what is in here."

THE AUDIT QUESTION FOR THIS STAGE

"Give me the component list for the version that was in production on the date of the incident."

The assessor wants that version's list rather than today's. If you cannot produce it, you cannot show whether you were affected, and the patch and reporting deadlines in Chapter 6 are already running. Build the SBOM to the deeper standard, keep one for every version, and add VEX before the findings pile up.

EVIDENCE AN ASSESSOR EXPECTS

- SBOM for every released version
- VEX statements
- Lifecycle data per component

Where this comes from. Regulation (EU) 2024/2847, Annex I Part II point (1), Article 13(24), Annex VII points 2(b) and 8. CISA, 2026 Minimum Elements for a Software Bill of Materials, version 2.1, 29 July 2026. PCI DSS v4.0.1, Requirements 6.3.2, 12.5.1 and 12.3.4. NIST SP 800-53 Rev. 5, control CM-8. Regulation (EU) 2022/2554, Article 8(2), (5) and (6); Commission Delegated Regulation (EU) 2024/1774, Articles 4 and 10(2). Implementing Regulation (EU) 2024/2690, Annex point 12.4; ENISA Technical Implementation Guidance, June 2025, section 12.4.

CHAPTER 6

How fast must you fix a vulnerability, and when must you report it?



THE SHORT VERSION

There are two kinds of clock and they are often confused. Patch clocks say how fast a vulnerability must be fixed. Reporting clocks say how fast an authority must be told about an exploited vulnerability or an incident. The CRA, NIS2 and DORA run both. FedRAMP's 2026 rules introduce their own incident-reporting clock. PCI DSS runs only a patch clock. The clocks start at different moments, from awareness of exploitation to the release of a patch, but none of them stops because the vulnerable code belongs to a project you do not control.

The regulations set two kinds of deadlines, and they are easy to confuse because both often run at once. One says how quickly the vulnerability must be fixed. The other says how quickly an authority must be told about it. Keeping the two apart matters because they are measured differently, start at different moments, and are enforced in different ways. Under PCI DSS or FedRAMP, a missed patch deadline usually surfaces as an assessment finding. Under the CRA, both are legal duties, and failing to remediate without delay falls in the same penalty tier as failing to report.

Patch clock

How fast the vulnerability must be fixed. Usually surfaces as an assessment finding under PCI DSS or FedRAMP.

Reporting clock

How fast an authority must be told about an exploited vulnerability or an incident. A legal duty under the CRA, NIS2, DORA, and a FedRAMP rule since 2026.

Patch deadlines in each regulation

The CRA does not set a number of days. It requires manufacturers to address and remediate vulnerabilities "without delay," to provide security updates free of charge, and to disseminate them without delay once available. "Without delay" is a standard an assessor will measure against your own remediation log and against what comparable manufacturers achieve.

FedRAMP used to set numbers, 30 days for high-severity findings, 90 for moderate and 180 for low, and many procedures still quote them. FedRAMP's 2026 rules replace them with a risk-based model that rates each vulnerability on its potential impact to the agency, whether it is likely to be exploited, and whether the affected component is reachable from the internet. The most dangerous combination is expected to be mitigated or remediated within two days at the Moderate baseline and within 12 hours at the High baseline. The rule is written as a SHOULD rather than a MUST, and a provider that departs from it has to explain why in its security documentation. The least dangerous tier that carries a deadline allows roughly six months.

The agencies that buy these services are on a clock of their own. CISA's Binding Operational Directive 26-04 of June 2026 gives them three calendar days for the most dangerous combinations of exposure, known exploitation, automatability (whether the exploit can be automated) and impact, 14 or 60 days for the rest, and the next scheduled upgrade for the least. FedRAMP has confirmed with CISA that providers following its 2026 rules meet those timelines, so the 12-hour and two-day windows, together with the due dates in CISA's KEV catalog, are the ones to plan against; section 8.2 has the directive's tiers.

PCI DSS still sets a number. Patches for vulnerabilities that your own risk ranking classifies as critical must be installed within one month of release. For everything else, you set the timeframe yourself, based on how critical the risk is to your environment; for the vulnerabilities ranked below high or critical, that timeframe has to rest on a documented targeted risk analysis.

DORA's technical standard sets no number of days. It requires weekly scanning of assets that support critical or important functions, every vulnerability recorded and tracked to resolution, deadlines for installing patches that you set yourself, with escalation when a deadline is missed, and an emergency patching procedure. NIS2's implementing rules require covered digital entities to address critical vulnerabilities without undue delay.

Incident and exploitation reporting deadlines

The CRA, NIS2 and DORA also require you to tell an authority, on short clocks.

Under the CRA, a manufacturer that becomes aware of an actively exploited vulnerability in its product must send an early warning within 24 hours, a fuller notification within 72 hours, and a final report no later than 14 days after a fix or mitigation is available. The clock starts when you become aware of the exploitation, which can be well before a CVE is published. The notifications go to the CSIRT designated as coordinator in the Member State where you have your main establishment, and to ENISA, through a single reporting platform. The obligation has applied since 11 September 2026 and it covers products already on the market.

PATCH CLOCKS AT A GLANCE

EU CRA	"Without delay"
FedRAMP	Risk-tiered: 12 hours at High and 2 days at Moderate for the most dangerous findings, up to 192 days for the lowest tier that carries a deadline; mandatory 7 December 2026. CISA BOD 26-04 sets 3, 14 or 60 calendar days for agencies, which FedRAMP's rules are designed to meet.
PCI DSS	One month for critical vulnerabilities; entity-set, risk-ranked schedule for the rest
NIS2	No number in the Directive; "without undue delay" for critical vulnerabilities under the implementing rules for covered entities
DORA	No number; weekly scanning, deadlines you set with escalation, emergency patching

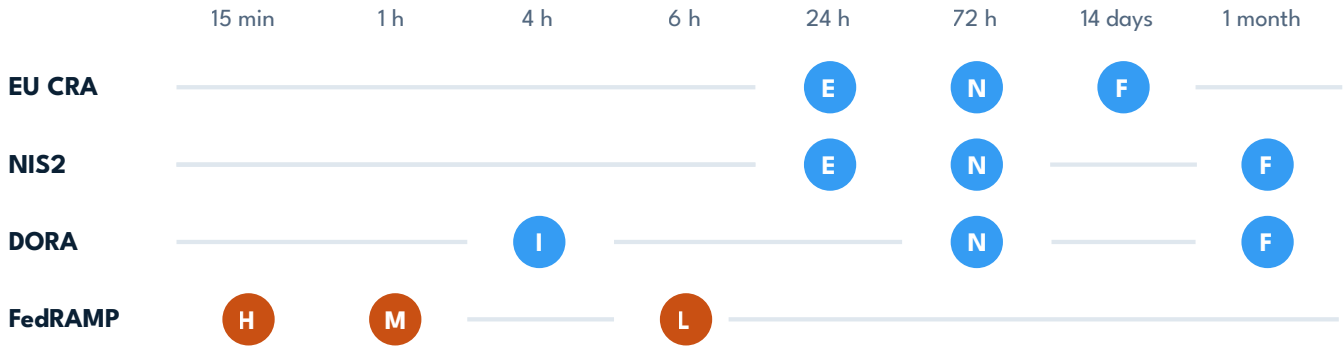
Under NIS2, an essential or important entity that suffers a significant incident must send an early warning within 24 hours, a notification within 72 hours, and a final report within one month of that notification.

Under DORA, a financial entity that classifies an incident as major must send an initial notification within four hours of the classification and no later than 24 hours after becoming aware of the incident, an intermediate report 72 hours after that, and a final report within one month of the intermediate report.

Under FedRAMP's 2026 rules, a provider that identifies an incident affecting, or likely to affect, the confidentiality or integrity of federal customer data must send an initial incident report to FedRAMP and the affected agencies. When the incident's potential agency impact is rated 3 or higher on FedRAMP's five-point scale, that report is due within six hours at the Low baseline, one hour at the Moderate baseline, and 15 minutes at the High baseline, followed by ongoing reports until the incident is resolved and a final report after recovery. The rules apply to new 20x certifications from 4 July 2026 and to Rev. 5 services from 1 January 2027.

These clocks measure how fast you report, and the patch clocks above measure how fast you fix. Both depend on the same evidence, because a report filed within 24 hours requires an inventory that can tell you within hours whether you are affected.

REPORTING CLOCKS AT A GLANCE



PCI DSS No reporting clock; patch clock only

E early warning **I** initial notification **N** notification or intermediate report **F** final report

H M L FedRAMP initial report at High, Moderate, Low

Clocks start at different moments. The CRA final report runs from the availability of a fix; NIS2 and DORA final reports run from the previous report. DORA's initial notification is due 4 hours after classification and no later than 24 hours after awareness. FedRAMP deadlines apply to incidents rated 3 or higher on its impact scale.

Why the deadlines apply to you and not to the maintainer

None of these texts has an exception for software you did not write. If the vulnerability is in a library four levels down your dependency tree, the CRA's 24-hour reporting clock, FedRAMP's 12-hour or two-day patch windows, and PCI's one month are still yours. A vulnerability in a transitive dependency is your finding.

That has a consequence most teams discover late. The patch clock assumes a patch exists. When the maintainer of a component has released a fix, your job is to ship it. When the maintainer has not, or no longer exists, the clock keeps running and the regulation still expects you to meet it. Chapter 7 covers that case; your remediation capability has to extend to components whose upstream does not respond.

What the remediation log must record

The artifact for this chapter is the remediation log, and the assessor will read it as a timeline. For each vulnerability it should carry the moment you learned of it, the triage decision and who made it, the moment a fix was available, the moment the fix was deployed, and for exploited vulnerabilities and incidents, the moments each report was filed.

Published advisories for the vulnerabilities you fix are a CRA requirement in their own right, and a contractual service level from whoever supplies your components is the evidence that the clock can be met for components you do not maintain yourself.

THE AUDIT QUESTION FOR THIS STAGE

"Show me the mean and the worst case time to remediate critical findings in this period, and the dates on which you reported the exploited ones."

Log both clocks separately, and make sure the inventory from Chapter 5 can tell you within hours which products a vulnerability affects.

EVIDENCE AN ASSESSOR EXPECTS

Remediation log with both clocks **Report filing dates**

Where this comes from. Regulation (EU) 2024/2847, Annex I Part II points (2), (4) and (8), Article 14, FedRAMP 2026 Consolidated Rules, Vulnerability Detection and Response, and Incident Evaluation and Communication; FedRAMP Notice 0014 on CISA Binding Operational Directive 26-04. PCI DSS v4.0.1, Requirements 6.3.1, 6.3.3 and 11.3.1.1. Directive (EU) 2022/2555, Articles 21(2)(e) and 23(4); Implementing Regulation (EU) 2024/2690, Annex points 6.6 and 6.10. Regulation (EU) 2022/2554, Articles 9(4)(f) and 19; Commission Delegated Regulation (EU) 2024/1774, Article 10; Commission Delegated Regulation (EU) 2025/301, Article 5.

CHAPTER 7

What must you do when a project stops maintaining a component you use?



THE SHORT VERSION

Your obligation to keep a component secure outlives the maintainer's willingness to do so. The CRA sets a minimum support period. FedRAMP gives you exactly two acceptable options for an unsupported component: replace it or secure continued support. PCI DSS wants a management-approved plan for every technology approaching end of life. DORA wants a yearly risk assessment of every legacy system, and NIS2's implementing rules, read with ENISA's guidance, point to an end-of-life date as a field for the asset inventory. An unmaintained component with no plan is the finding every assessor is trained to look for.

Open source projects stop being maintained: maintainers move on, a major version replaces the one you built on, a company that funded the work changes its strategy. What is new is that the regulations now assign the consequence to you.

The CRA support period continues after upstream support ends

The CRA requires a manufacturer to declare a support period for each product. It must be at least five years unless the product's expected use is shorter. For the whole of that period the vulnerability-handling obligations apply: identify, remediate without delay, distribute fixes securely, disclose what you fixed. The regulation does not care whether a component inside the product is still maintained by anyone. The product and the support period are yours, and the duty to handle vulnerabilities runs to the end of that period.

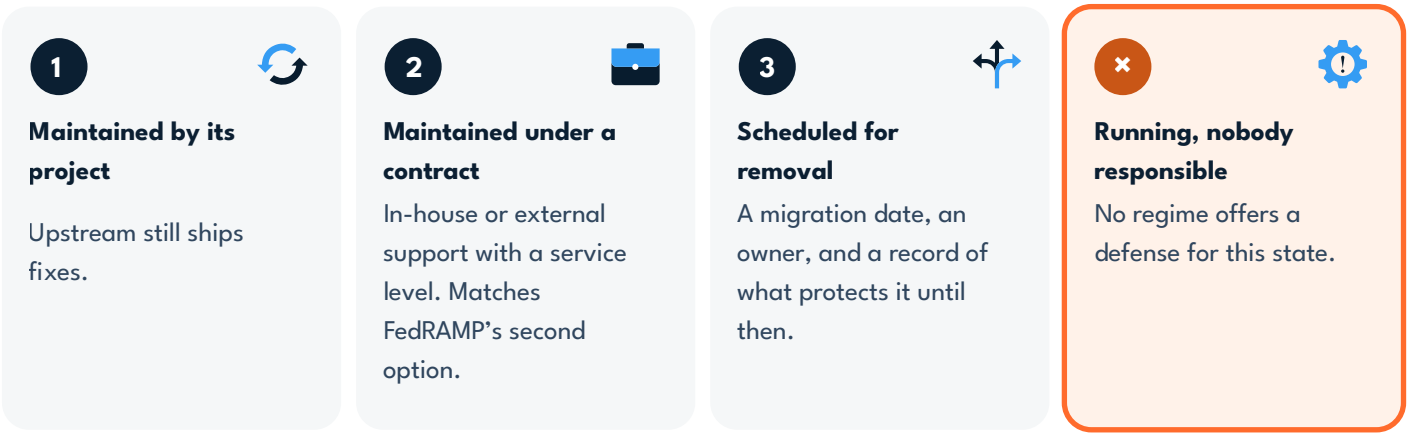
This is the point where a manufacturer discovers how much of its product's security posture has been outsourced to volunteers.

FedRAMP states the choice more bluntly than anyone. Its control on unsupported system components, which sits in every baseline, gives a cloud service provider two options when a component is no longer supported by its developer, vendor or manufacturer: replace it, or provide an alternative source of continued support, in-house or from an external provider. The control's discussion says such contractual relationships can include open source software value-added vendors.

A practical model for unsupported components

Building on that, this guide uses a three-state model: a component is maintained by its project, maintained under a contract, or scheduled for removal with a date. The state to avoid is a fourth one, common across large dependency trees, where the component is still running and nobody is responsible for it. The third state needs a migration date, an owner, and a record of what protects the component until then; "we plan to migrate" on its own does not qualify.

THE THREE ACCEPTED STATES, AND THE ONE TO AVOID



State 1 is a component that is still supported. States 2 and 3 match FedRAMP's two options: alternative support, and replacement while it is in progress.

DORA, PCI DSS and NIS2 requirements for end-of-life components

DORA requires a specific risk assessment of every legacy ICT system at least yearly and defines a legacy system as one that has reached end of life, cannot be upgraded or fixed, or is no longer supported by its supplier or a third-party provider, but is still in use; its technical standard requires asset records to carry support end dates.

A DORA-supervised bank is expected to know when each asset's support ends and to reassess every asset past that date yearly.

PCI DSS asks for something similar every year: a review of the hardware and software technologies in use that confirms they still receive security fixes from their vendors promptly, documents any vendor announcement of end of life, and records a plan, approved by senior management, to remediate the outdated ones.

NIS2's implementing rules, for the digital entities they cover, require a complete and current asset

inventory, and the regulation's recital and ENISA's guidance on those rules list each asset's end of life among the fields to consider. The Directive itself is silent on end of life.

What the CRA ten-year rule actually requires

A good deal of writing about the CRA misreads its ten-year rule. The regulation says that each security update, once issued, must remain available for at least ten years or for the rest of the support period, whichever is longer. That is a duty to keep issued updates downloadable; the duty to produce new fixes runs for the support period, which must be at least five years.

If a vendor tells you the CRA requires ten years of patching, they have read it wrong, and if your own documentation says so, correct it, so that you do not plan and budget against a requirement that does not exist.

What to record for components past end of life

The artifact at this stage is a register of components approaching or past their upstream's end of life, with dates and owners. For each component already past that date, the register points to the evidence that it is in one of the accepted states: a support agreement with a service level, backed by a remediation log that continues after the upstream stopped, or a dated removal plan. For each component still approaching that date, it records the date and the decision taken. Time-bound risk acceptance records cover anything you have decided to carry for a while. The remediation log from Chapter 6 should not have a gap that begins on the day the upstream project stopped.



THE AUDIT QUESTION FOR THIS STAGE

"This component's upstream ended support two years ago. What is your basis for still running it?"

Make sure every component in your inventory has one of the three accepted answers, and that the inventory from Chapter 5 can tell you which components need the question asked.

EVIDENCE AN ASSESSOR EXPECTS

- Support agreement with SLA
- Dated removal plan
- Time-bound risk acceptance
- End-of-life register

Where this comes from. Regulation (EU) 2024/2847, Article 13(8) and 13(9). NIST SP 800-53 Rev. 5, control SA-22 (Low, Moderate and High baselines). PCI DSS v4.0.1, Requirement 12.3.4. Regulation (EU) 2022/2554, Article 8(7); Commission Delegated Regulation (EU) 2024/1774, Article 4. Implementing Regulation (EU) 2024/2690, Annex point 12.4; ENISA Technical Implementation Guidance, June 2025, section 12.4. NIST Cybersecurity Framework 2.0, subcategory PR.PS-02.

CHAPTER 8

Which regulations apply to you, and which of their clauses reach your open source components?

THE SHORT VERSION

This chapter contains the clause numbers. Each of the five regimes has its own section: who it applies to, when it took effect, what happens on failure, what changed in the last twelve months, a table that places its clauses on the five lifecycle stages from Chapters 3 to 7, and the questions its assessor is likely to ask about open source. A closing note covers the voluntary frameworks auditors use as yardsticks.

Chapters 3 to 7 followed a package through its life and named the regulations as they came up. This chapter is organized the other way, by regulation, for readers who arrived with one regime in mind: find your section and work back to the chapters it points to.

In the tables, the "what it says" column quotes the text, trimmed to the words that carry the requirement where a clause is long. The "what it means for your packages" column is the translation from Chapter 2. The "evidence" column names which of the ten artifacts described in Chapter 2 proves the requirement is met. Where a regime is silent at a stage, the table says so rather than stretching a clause to fit.

8.1 The EU Cyber Resilience Act

APPLIES TO

Manufacturers, importers and distributors of products with digital elements made available on the EU market, wherever in the world they are based. Hardware and software alike. Open source software is in scope when supplied in the course of a commercial activity; a manufacturer who integrates open source components into a commercial product is responsible for them regardless. Non-commercial projects are outside the manufacturer obligations, and a lighter category of "open source software steward" exists for foundations that support them.

FROM WHEN

In force since 10 December 2024. Reporting obligations for actively exploited vulnerabilities and severe incidents have applied since 11 September 2026, including to products already on the market. All other manufacturer obligations, conformity assessment and CE marking apply from 11 December 2027.

PENALTIES

Up to EUR 15 million or 2.5 percent of worldwide annual turnover, whichever is higher, for failing the essential requirements or the manufacturer's core obligations.

WHAT CHANGED IN THE LAST TWELVE MONTHS

The Commission published its non-binding CRA guidance, C(2026) 5252, on 27 July 2026. It clarifies when open source counts as "placed on the market," describes steward obligations, and confirms that due-diligence, upstream-reporting and fix-sharing duties apply to any integrated open source component. Reporting went live on 11 September 2026. No implementing act on SBOM format has been adopted, so "top-level dependencies" remains the only formal depth requirement.

Stage and clause	What it says	What it means for your packages	Evidence
Adoption Art. 13(5); Commission guidance	Manufacturers "shall exercise due diligence when integrating components sourced from third parties," including free and open source software not supplied commercially	A documented, risk-proportionate reason to trust each component before it is integrated	**Component intake record; source policy; screening result**
Integrity Annex I Part I 2(f); Part II (7)	Protect the integrity of "commands, programs and configuration against any manipulation or modification not authorised by the user"; "provide for mechanisms to securely distribute updates"	Verify that the package installed is the one published; distribute your own updates so they cannot be tampered with in transit	**Signature verification log; provenance attestation**
Inventory Annex I Part II (1); Art. 13(24); Annex VII 2(b) and 8	SBOM "in a commonly used and machine-readable format covering at the very least the top-level dependencies"; Commission may specify format by implementing act; SBOM is part of the technical documentation and disclosable to a market surveillance authority on reasoned request	A per-release SBOM, kept with the technical file; the floor is direct dependencies, the working standard is the full graph	**SBOM per release; VEX**
Remediation Annex I Part I 2(a); Part II (2), (4), (8); Art. 14	Made available "without known exploitable vulnerabilities"; "address and remediate vulnerabilities without delay, including by providing security updates"; "share and publicly disclose information about fixed vulnerabilities"; updates "disseminated without delay" and "free of charge"; early warning "within 24 hours," notification "within 72 hours," final report "no later than 14 days after a corrective or mitigating measure is available"	Every vulnerability in every component, including transitive ones, is yours to fix and, if exploited, to report	**Remediation log; published advisories; reporting records**
Support period Art. 13(8), 13(9)	Vulnerabilities "of that product, including its components, are handled effectively" for the support period, which "shall be at least five years" unless the product is expected to be in use for less; each issued security update "remains available … for a minimum of 10 years or for the remainder of the support period, whichever is longer"	Fixes for every component must continue for the declared period, whether or not the upstream project does; issued fixes stay downloadable	**Support agreement with SLA; EOL register; remediation log past upstream EOL**

QUESTIONS A CRA ASSESSOR WILL ASK ABOUT OPEN SOURCE

- What is your documented basis for each third-party component in the product?
- Can you produce the SBOM for the version a customer is running, and for a vulnerability in a transitive dependency, how quickly can you tell whether that version is affected?
- For each component whose project has stopped, who is producing its security updates for the rest of your declared support period?

Primary texts. Regulation (EU) 2024/2847, Articles 13, 14, 24, 64 and 71, Annex I and Annex VII. European Commission guidance C(2026) 5252, 27 July 2026.

8.2 FedRAMP

APPLIES TO

Cloud services sold to US federal agencies. Authorization is granted against security baselines built from NIST SP 800-53 Revision 5, at Low, Moderate or High impact, and maintained through continuous monitoring, with assessments by an accredited third-party assessment organization, or 3PAO. A control "in the baseline" is one every authorized service at that level must implement.

FROM WHEN

The Rev. 5 baselines have applied since 2023. FedRAMP's 2026 consolidated rules on vulnerability detection and response became optional on 4 July 2026 and mandatory for all authorized services on 7 December 2026, with a grace period ending 7 March 2027.

CONSEQUENCE OF FAILURE

FedRAMP does not fine; it withdraws or declines authorization, which ends the service's eligibility for federal use.

WHAT CHANGED IN THE LAST TWELVE MONTHS

The fixed remediation windows of 30, 90 and 180 days were replaced by a risk-based model that rates each vulnerability on potential agency impact, likely exploitability and internet reachability, with deadlines from 12 hours at High, or two days at Moderate, to roughly six months.

CISA's Binding Operational Directive 26-04 of 10 June 2026 reprioritized federal patching around exposure, KEV status, automatability and impact, and FedRAMP's response folds those priorities into its rules. The directive gives agencies three calendar days for the most dangerous combinations, meaning a KEV-listed vulnerability on a publicly exposed asset (unless the exploit is neither automatable nor gives total control), or an automatable exploit that gives total control where

the asset is publicly exposed or the vulnerability is KEV-listed. It gives 14 or 60 days for the rest, and for vulnerabilities that are not exposed, not exploited and not automatable, a fix is mandated at the next scheduled system upgrade. The clock starts when CISA adds the vulnerability to the KEV catalog or the agency detects it on an asset, whichever comes first.

3 days
Most exposed KEV-listed or automatable, total-control vulnerabilities

14 days
Next tier

60 days
Remaining tier

Next upgrade
Not exposed, not exploited, not automatable

CISA BOD 26-04 deadlines for federal agencies, which FedRAMP providers following the 2026 rules are designed to meet.

The rules also treat a failure of the detection process itself as a vulnerability to be remediated. Detection cadences tighten as well. For Rev. 5 authorizations the required floor stays monthly, with sampling of representative resources recommended as often as every three days in the stricter classes, or daily at the highest; on FedRAMP's 20x track, verification at least every seven days in Class B and every three days in Class C is mandatory.

Stage and clause	What it says	What it means for your packages	Evidence
Adoption SR-3, SR-5, SA-4, SI-3, SR-11 (all baselines)	"Identify and address weaknesses or deficiencies in the supply chain elements" (SR-3); "employ … acquisition strategies, contract tools, and procurement methods" against supply chain risks (SR-5); acquisition contracts "include … security and privacy functional requirements" (SA-4); "malicious code protection mechanisms at system entry and exit points" (SI-3); "detect and prevent counterfeit components from entering the system" (SR-11)	A documented process for how components enter the system, with each one screened for malicious code and for names that imitate real packages	**Intake record; source policy; screening result**
Integrity SI-7 (Moderate, High); CM-14 (High)	"Employ integrity verification tools to detect unauthorized changes to the following software, firmware, and information" (SI-7); "prevent the installation of" components "without verification that the component has been digitally signed using a certificate that is recognized and approved by the organization" (CM-14)	Provenance (SR-4) is in no baseline, so it is an expected practice; signature verification is a requirement at High and an expected practice below it	**Verification log; provenance attestation**

Stage and clause	What it says	What it means for your packages	Evidence
Inventory CM-8 (all); VDR-CSO-FAV	"Develop and document an inventory of system components that" "accurately reflects the system" and "includes all components within the system" (CM-8); "providers must treat problems or failures with their vulnerability detection and response processes as vulnerabilities"	A component inventory current enough to drive the new scanning cadences	**SBOM per release; VEX**
Remediation SI-2, RA-5 (all); 2026 VDR and VER rules; BOD 26-04	Flaw remediation (SI-2) and vulnerability scanning (RA-5); providers "SHOULD partially mitigate vulnerabilities, fully mitigate vulnerabilities, or remediate vulnerabilities … within the maximum timeframes": 12 hours (High) or 2 days (Moderate) for a likely-exploitable, internet-reachable finding of the highest impact, up to 192 days for the lowest tier; BOD 26-04: 3, 14 or 60 days; mandatory 7 December 2026	Every finding in every component falls under a timeframe you are expected to meet, or to justify missing, however deep the component sits	**Remediation log with timestamps; SLA from component supplier**
Support period SA-22 (all baselines)	"Replace system components when support for the components is no longer available from the developer, vendor, or manufacturer," or "provide … alternative sources for continued support," in-house or external; discussion: "Such contractual relationships can include open-source software value-added vendors"	An unsupported open source component is either replaced or given an alternative source of support, in-house or under contract; nothing else is an accepted state	**Support agreement; EOL register; removal plan**

QUESTIONS A 3PAO WILL ASK ABOUT OPEN SOURCE

- Which of your components are past their upstream's support, and for each one, is it being replaced or who is supporting it?
- When a likely-exploitable, internet-reachable vulnerability appears in a transitive dependency, can your inventory tell you within hours whether you are affected, given that the deadline may be 12 hours or two days?
- How do you verify that the components in your build are authentic and unaltered?

Primary texts. NIST SP 800-53 Rev. 5 and the FedRAMP baseline allocations for controls SA-4, SA-22, SI-2, SI-3, SI-7, RA-5, CM-8, CM-14, SR-3, SR-4, SR-5 and SR-11. FedRAMP 2026 Consolidated Rules, Vulnerability Detection and Response, and Incident Evaluation and Communication; FedRAMP Notice 0014 on CISA Binding Operational Directive 26-04.

8.3 PCI DSS version 4.0.1

APPLIES TO

Every entity that stores, processes or transmits payment card data, and the service providers that can affect its security. Enforced contractually through the card brands and acquiring banks, and assessed by Qualified Security Assessors (QSAs) or by self-assessment, depending on the card brands' programs.

FROM WHEN

Version 4.0 was published in 2022 and version 4.0.1 in June 2024. The future-dated requirements of 4.0, including the inventory of third-party software components, became mandatory on 31 March 2025.

CONSEQUENCE OF FAILURE

Non-compliance findings in the Report on Compliance, contractual penalties set by the acquirer or card brand, and in serious cases loss of the ability to process card payments.

WHAT CHANGED IN THE LAST TWELVE MONTHS

Nothing in the text, but assessors have moved from noting the version 4 requirements to scoring them. The component inventory and the payment-page script controls are now tested as a matter of course.

Stage and clause	What it says	What it means for your packages	Evidence
Adoption 5; 6.3.2	"Protect All Systems and Networks from Malicious Software" (Requirement 5); "An inventory of bespoke and custom software, and third-party software components incorporated into bespoke and custom software is maintained to facilitate vulnerability and patch management" (6.3.2)	Components are screened for malicious software before they enter the system, and the inventory starts when a component is adopted and is kept current from then on	**Screening result; intake record feeding the SBOM**
Integrity 6.4.3; 11.6.1	For payment page scripts, "a method is implemented to confirm that each script is authorized," "… to assure the integrity of each script," "an inventory of all scripts is maintained"; "a change- and tamper-detection mechanism is deployed" to alert personnel to "unauthorized modification"	Know and verify every script and component in the payment path; detect when one changes	**Script inventory; integrity verification log; tamper-detection alerts**

Stage and clause	What it says	What it means for your packages	Evidence
Inventory 6.3.2; 12.5.1; 12.3.4	Component inventory (6.3.2, quoted above); "an inventory of system components that are in scope for PCI DSS, including a description of function/use, is maintained and kept current" (12.5.1); "hardware and software technologies in use are reviewed at least once every 12 months" (12.3.4)	A per-release component list for every in-scope application, usable for patching	**SBOM per release; asset inventory**
Remediation 6.3.1; 6.3.3; 11.3.1.1	"New security vulnerabilities are identified using industry-recognized sources"; third-party software "… are covered"; "patches/updates for critical vulnerabilities … are installed within one month of release"; all others "within an appropriate time frame as determined by the entity's assessment of the criticality of the risk"; lower-ranked vulnerabilities "addressed based on the risk defined in the entity's targeted risk analysis"	Every component vulnerability is risk-ranked and the critical ones are fixed within a month, including transitive ones; the rest on a schedule you set and justify	**Remediation log; risk analysis records**
Support period 12.3.4	Technologies "reviewed at least once every 12 months," including "analysis that the technologies continue to receive security fixes from vendors promptly," documentation of vendor "end of life" plans, and "a plan, approved by senior management, to remediate outdated technologies"	Every component past or approaching its upstream's end of life has a management-approved plan	**EOL register; approved remediation plan; support agreement**

QUESTIONS A QSA WILL ASK ABOUT OPEN SOURCE

- Show me the inventory of third-party components in this payment application and tell me how it is kept current.
- Show me a critical vulnerability from the last twelve months and the date its fix was installed.
- Show me the annual technology review and the plan for the components whose vendors have announced end of life.

Primary texts. PCI DSS v4.0.1, Requirements 5, 6.3.1, 6.3.2, 6.3.3, 6.4.3, 11.3.1.1, 11.6.1, 12.3.4 and 12.5.1.

8.4 NIS2

APPLIES TO

Essential and important entities across eighteen sectors in the EU, from energy and transport to digital infrastructure and public administration, under national laws that transposed the Directive. For a defined set of digital entities (cloud, data centers, CDNs, managed and managed security service providers, online marketplaces, search engines, social platforms, trust services, DNS and TLD registries) an implementing regulation of October 2024 sets detailed technical requirements. ENISA's guidance is written for those entities; for the rest it is a reference.

FROM WHEN

Through national law since 18 October 2024 in the Member States that transposed on time; several transposed later and their laws apply from their own dates. The implementing regulation for digital entities was adopted on 17 October 2024 and applies alongside those laws.

PENALTIES

Essential entities: up to EUR 10 million or 2 percent of worldwide annual turnover. Important entities: up to EUR 7 million or 1.4 percent. Management bodies must approve the risk management measures, oversee them, and can be held liable.

WHAT CHANGED IN THE LAST TWELVE MONTHS

Nothing in the Directive's text. Supervision under the national laws is under way, and ENISA's guidance of June 2025 has become the working reference for the evidence covered entities are expected to show. Where this guide attributes a field such as an end-of-life date to NIS2, the requirement is the implementing regulation's and the field comes from its recital and ENISA's guidance.

Stage and clause	What it says	What it means for your packages	Evidence
Adoption Art. 21(2)(d), (e); 21(3); CIR Annex 5.1.1, 5.1.2, 5.2, 6.9.1, 6.9.2	The Directive lists "supply chain security, including … the relationships between each entity and its direct suppliers" and "security in network and information systems acquisition, development and maintenance," and tells entities to "take into account the vulnerabilities specific to each direct supplier" and "their secure development procedures." The implementing regulation requires a "supply chain security policy," "a registry of their direct suppliers and service providers," and "measures that detect or prevent the use of malicious or unauthorised software"	An open source project is not a supplier under contract, but the same questions apply: how is it maintained, how are vulnerabilities handled, and is that written down before you adopt it; covered digital entities also screen packages so malicious or look-alike ones stay out	**Intake record; supplier assessment; source policy; screening result**

Stage and clause	What it says	What it means for your packages	Evidence
Integrity CIR Annex 6.6.1(c)	Neither text requires signature or provenance verification. The closest clause is the patch management rule: “security patches come from trusted sources and are checked for integrity”	Patches you apply must be checked for integrity; beyond that, signature and provenance verification is good practice rather than a requirement	**Verification log and provenance attestation, as good practice**
Inventory CIR Annex 12.4.1, 12.4.2; ENISA guidance	No inventory clause in the Directive. Covered digital entities “shall develop and maintain a complete, accurate, up-to-date and consistent inventory of their assets”; ENISA lists “date of asset’s last update/patch” and “asset end of life, where applicable” among the fields to consider	A component inventory that carries lifecycle data: when each package was last patched and when its project stops maintaining it	**SBOM combined with lifecycle data**
Remediation CIR Annex 6.6.1, 6.10.2; Art. 23(4)	No patch deadline in the Directive. Patches “applied within a reasonable time after they become available”; “address, without undue delay, vulnerabilities identified … as critical”; early warning “within 24 hours,” notification within “72 hours,” final report “not later than one month after the submission of the incident notification”	Every vulnerability is tracked to resolution, critical ones without undue delay; a significant incident involving a component is reported on the Directive’s clocks	**Remediation log; reporting records**
Support period CIR Annex 12.4; ENISA guidance; CIR Annex 6.6.1(d)	The Directive is silent on end of life. The implementing regulation requires the asset inventory above, ENISA adds "asset end of life" to it, and the patch rules require "additional measures are implemented and residual risks are accepted in cases where a patch is not available"	Each component past its upstream’s end of life is supported under a contract or carried under a documented risk acceptance with additional measures	**EOL register; support agreement; risk acceptance records**

QUESTIONS A NIS2 SUPERVISOR WILL ASK ABOUT OPEN SOURCE

- How did you assess the secure development practices of the projects and suppliers your critical systems depend on?
- Can your asset inventory show, for each component, when it was last patched and when its support ends?
- For the components whose upstream has stopped, what additional measures are in place and who accepted the residual risk?

Primary texts. Directive (EU) 2022/2555, Articles 21, 23 and 34. Commission Implementing Regulation (EU) 2024/2690, Article 1 and Annex points 5.1, 5.2, 6.6, 6.9, 6.10 and 12.4. ENISA Technical Implementation Guidance, June 2025, sections 5.1, 5.2, 6.6, 6.9, 6.10 and 12.4.

8.5 DORA

APPLIES TO

Financial entities in the EU, from banks and insurers to payment and crypto-asset service providers, and ICT third-party service providers designated as critical under Article 31. ICT risk management requirements sit in a 2024 regulatory technical standard; incident reporting deadlines in a second delegated regulation published in February 2025.

FROM WHEN

In application since 17 January 2025. The ICT risk management standard (Delegated Regulation 2024/1774) has been in force since 15 July 2024; the incident reporting standard (Delegated Regulation 2025/301) since 12 March 2025.

PENALTIES

Administrative penalties and remedial measures are set and applied by national competent authorities; critical ICT third-party providers face a separate oversight regime at EU level.

WHAT CHANGED IN THE LAST TWELVE MONTHS

Nothing in the text. The first year turned the register of information, the record of every ICT contract a financial entity holds, into a live supervisory tool, so a commercial supplier of maintained open source packages now appears in a register supervisors read.

Why it matters for open source. DORA's technical standard names open source in two of its articles and, unlike the CRA, which names free and open source software mainly to settle whose responsibility it is, attaches operational requirements to it: source code from open source projects must, where feasible, be analyzed and tested before production, and the usage of open source libraries in ICT services supporting critical or important functions must be tracked.

Stage and clause	What it says	What it means for your packages	Evidence
Adoption RTS Art. 16(2), 16(3), 16(8)	"Testing and approval of all ICT systems prior to their use"; source code reviews covering both static and dynamic "testing"; "where feasible, source code coming from open source projects" is "analysed and tested" before "deployment in the production environment"	Where feasible, every open source package is reviewed and tested before production, proportional to the criticality of the system, and the record of that review exists	**Intake record; test and review records**

Stage and clause	What it says	What it means for your packages	Evidence
Integrity RTS Art. 16(1)(c); Art. 9(4)(e)	Measures to "mitigate the risk of unintentional alteration or intentional manipulation of the ICT systems" during development, maintenance and deployment; "documented policies, procedures and controls for ICT change management, including changes to software, hardware, firmware components"	Verify that the package deployed is the package approved, and record every change to it	**Verification log; change records**
Inventory Art. 8(6); RTS Art. 10(2)(d); RTS Art. 4(2)(b)(ix)	"Maintain relevant inventories and update them periodically"; "track the usage of third-party libraries, including open-source libraries"; asset records carry "the end dates of the ICT third-party service provider's regular, extended, and custom support services"	A component inventory for every system supporting a critical or important function, with the support end date of each component	**SBOM combined with lifecycle data**
Remediation Art. 9(4)(f); RTS Art. 10(2), 10(4); Del. Reg. 2025/301 Art. 5	"Documented policies for patches and updates"; scanning "on at least a weekly basis" for assets supporting critical or important functions; "recording of any detected vulnerabilities … and the monitoring of their resolution"; patch "deadlines … and escalation procedures"; "emergency procedures"; major incident reporting "within four hours from the classification" and "no later than 24 hours" from awareness, intermediate "within 72 hours," final "no later than one month" after it	Every component vulnerability is recorded and tracked against deadlines you have set, and a major incident involving one is reported on the clocks above	**Remediation log; reporting records**
Support period Art. 3(3); Art. 8(7); RTS Art. 4	A legacy ICT system includes one "no longer supported by its supplier or by an ICT third-party service provider, but that is still in use"; entities "shall … at least yearly, conduct a specific ICT risk assessment on all legacy ICT systems"	Each component past its upstream's support is under a support contract or treated as a legacy system with a yearly risk assessment	**EOL register; legacy risk assessments; support agreement**

QUESTIONS A DORA SUPERVISOR WILL ASK ABOUT OPEN SOURCE

- Show me the record that this open source component was analyzed and tested before it went into production.
- Which open source libraries do your critical systems use, and when does each one's support end?
- For the legacy systems in your register, where is this year's risk assessment, and who provides their security fixes?

Primary texts. Regulation (EU) 2022/2554, Articles 3, 8, 9, 19 and 28. Commission Delegated Regulation (EU) 2024/1774, Articles 4, 10 and 16. Commission Delegated Regulation (EU) 2025/301, Article 5.

8.6 SOC 2, ISO 27001 and CIS Controls

These are not regulations, but they shape how auditors read the ones that are, and if you sell software to other businesses you will meet at least one of them every year. None adds a new obligation to the five regimes above; each is the form in which a customer's auditor will ask you the same questions.

SOC 2

CC7.1–CC7.5

Vulnerability detection, monitoring and response are tested against your own stated service levels, so an open source finding that cannot be closed becomes an exception in a report every prospect reads.

ISO/IEC 27001:2022

Annex A 8.8

Technical vulnerability management expects timely information, evaluation and "appropriate measures." A risk acceptance with no owner, no mitigation and no end date is not one.

CIS Controls v8.1

Controls 2 and 7

Control 2 names unsupported software in the software asset inventory; Control 7 expects the vulnerabilities in it to be remediated.

CHAPTER 9

What evidence must you be able to show an assessor?

THE SHORT VERSION

The ten artifacts below cover the five regimes in this guide. Most of them are generated by the pipeline, the registry or the scanner rather than written by a person, so the evidence accumulates as you work and does not have to be assembled before an audit.

Chapters 3 to 7 each ended with the question an assessor asks and the artifact that answers it. This chapter collects them into one list, says who owns each and when it is produced. It ends with a self-assessment you can fill in on one page.

The ten artifacts, their owners and their cadence

Artifact	Regimes it serves	Owner · cadence	Form an assessor accepts
1. Component source policy Adoption	CRA, NIS2, DORA, FedRAMP	Security · reviewed yearly	Short written policy naming approved sources and screening rules
2. Component intake record Adoption	CRA, DORA, NIS2, PCI DSS	Engineering · per component, at adoption	Structured record: origin, maintainer, license, advisories, open CVEs at adoption, decision
3. Pre-ingestion screening result Adoption	CRA, FedRAMP, PCI DSS, NIS2	Security (registry) · per package version	Registry admission criteria covering known malware, look-alike names and what the code would do when run; or, without a screening registry, a malware scan record per version
4. Signature verification log Integrity	CRA, FedRAMP, PCI DSS, DORA	Engineering (pipeline) · per build	Pipeline log showing each artifact's signature checked
5. Provenance attestation Integrity	CRA, FedRAMP (practice)	Supplier or pipeline · per artifact	Signed attestation with the SLSA level stated accurately
6. SBOM per release, with lifecycle data Inventory, Support period	All five	Engineering (pipeline) · per release, retained per version	CycloneDX or SPDX, generated from the build, full graph, with each component's upstream support status and end-of-life date
7. VEX statements Inventory, Remediation	All five (practice)	Security · as findings are triaged	Machine-readable status per vulnerability per product
8. Remediation log Remediation	All five	Security · per vulnerability	Timestamps: detected, triaged, fix available, deployed, reported where required
9. Support agreement with service level Support period	CRA, FedRAMP, PCI DSS, DORA	Procurement and Compliance · per contract	Contract naming the components covered and the patch commitment by severity
10. Risk acceptance record and removal plan Support period	All five	Compliance, approved by management · per exception, time-bound	Owner, end date, mitigation, approval; for removals, a dated plan

Self-assessment by lifecycle stage

For each of the five stages, pick one of three states: current, meaning the evidence exists and covers the versions now in production; stale or partial, meaning it exists but is out of date, incomplete or cannot be tied to a version; or missing.

Stage	Current	Stale or partial	Missing	Notes
Adoption. Can you show how each component was vetted before it was integrated?	☐	☐	☐	
Integrity. Can you show that the artifact in production is the one the publisher released?	☐	☐	☐	
Inventory. Can you produce the full component list for the version in production on a given date?	☐	☐	☐	
Remediation. Can you show time to remediate by severity, and the dates of any required reports?	☐	☐	☐	
Support period. Can you show, for each component past its upstream's support, which of the three accepted states it is in?	☐	☐	☐	

Reading your result. A common pattern is "missing" in the first two rows, "stale or partial" in the other three. The inventory is usually a scanner output or a top-level list rather than a full SBOM kept for each version, and the remediation log inherits that gap: it is complete for the components the scanner could see and silent on the transitive dependencies it could not. If that is your result, Chapters 3, 4 and 5 are where to start, and Chapter 7 follows once the inventory can show which components have lost their upstream.

CHAPTER 10

Where SecureChain and ELS fit in the lifecycle

THE SHORT VERSION

TuxCare's SecureChain and Endless Lifecycle Support cover the package lifecycle this guide describes, from a screened and attested source at adoption to security fixes that continue after a project's own maintainers have stopped. They supply most of the package-level artifacts in Chapter 9, and they are built to support the policy and risk decisions the regulations assign to you.

This guide has been deliberately neutral about products until now, and this chapter is short. It describes what TuxCare provides at each stage of the lifecycle, in the same terms the previous chapters used, so that you can judge for yourself which of your gaps it closes.

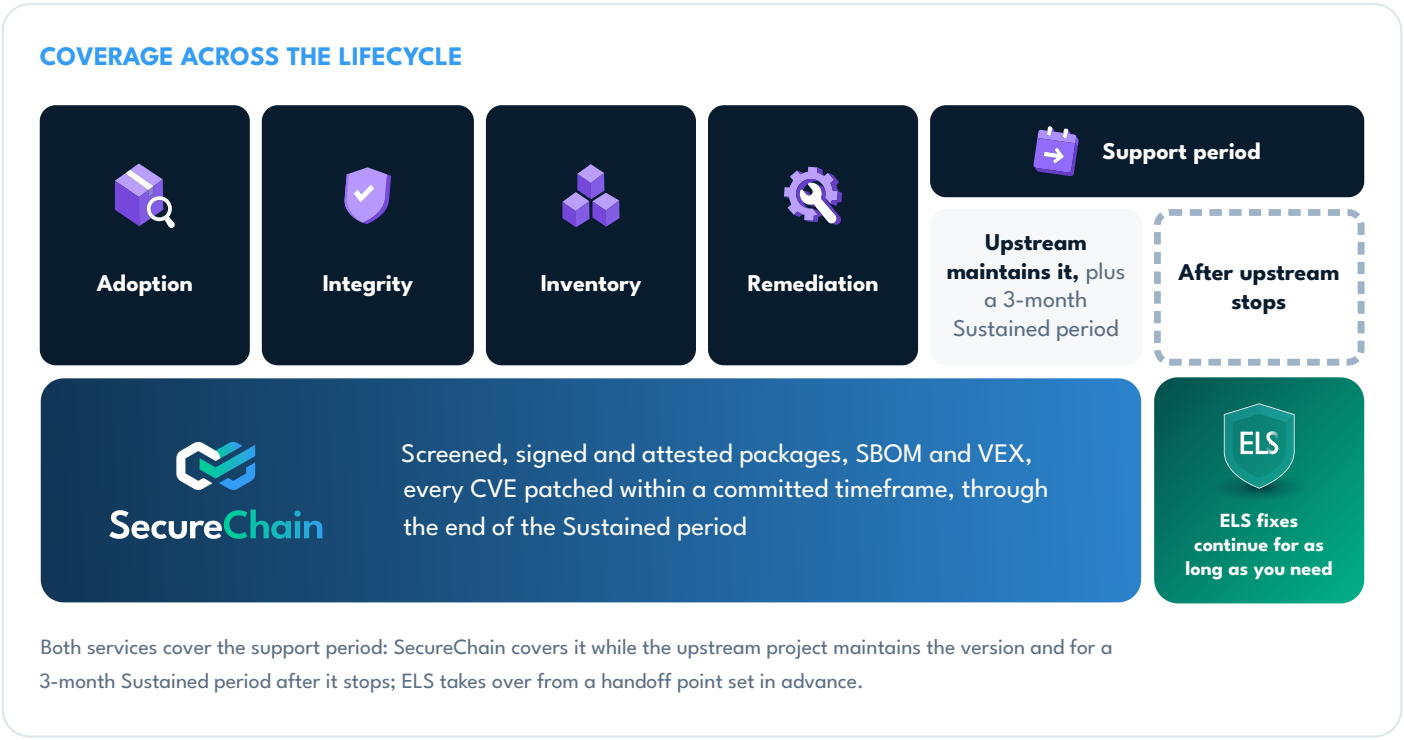
SecureChain is a curated source of open source packages that TuxCare screens, builds, signs, attests and keeps patched, delivered to subscribers through a private registry. Endless Lifecycle Support is the complementary service that continues to produce security fixes for a package version after its own project has stopped. Together they are one commitment: a package that enters your environment through SecureChain was trusted on the day it arrived and is kept secure for as long as you run it.

What SecureChain and ELS provide at each stage

At adoption, every package is screened before it enters the catalog: for known malware, for names that imitate popular packages, and for what its code would do once installed and run. Only packages that pass are admitted to the registry, which gives your policy a single source to name: packages come from one place, and that place has rules.

At integrity, every package is signed and built to support verifiable SLSA Level 3 provenance, so that "was this built from the source the project published?" is answered by the provenance, and "was it altered between the registry and my build?" by verifying the signature.

At inventory, packages come with machine-readable SBOMs and VEX statements in CycloneDX format covering their full dependency graphs. The VEX statements record, for each known vulnerability in a package, whether it has



Both services cover the support period: SecureChain covers it while the upstream project maintains the version and for a 3-month Sustained period after it stops; ELS takes over from a handoff point set in advance.

been fixed in the version supplied or still affects it. The registry records which package versions are in use across its customer base, so when a CVE is published TuxCare can see which versions in use are affected and prioritize the fix.

At remediation, every CVE is patched within a committed timeframe based on its severity. When the upstream project has released a fix, SecureChain rebuilds the fixed version. When the upstream has not, or no longer responds, TuxCare patches the package itself, preserving compatibility with the version you run, and shares the fix upstream where the project still maintains that version. Each fix comes with an updated VEX statement marking the vulnerability as fixed in the new version.

At end of life, the point at which coverage passes from SecureChain to ELS is set in advance. A package version stays under SecureChain's coverage for a Sustained period of three months after its upstream stops fixing it, which TuxCare defines as a missed CVE SLA or nine months without a release, and then Endless Lifecycle Support takes over for as long as you need it, covering the whole dependency tree of the version you run. When a buried dependency is patched, everything above it is rebuilt on the fix. The remediation log does not stop on the day the project did.

Which Chapter 9 artifacts a subscription supplies

Of the ten artifacts, a SecureChain subscription supplies the provenance attestation, the SBOM per package and the VEX statements directly, and it is the support agreement with a service level for every package it covers. It gives your pipeline the signatures to verify and the lifecycle data to add to your inventory. It does not hand you a screening result: its registry admits only packages that passed screening, so your source policy can name it as an approved source.

A SUBSCRIPTION SUPPLIES

- A registry that admits only screened packages
- Provenance attestation and signatures to verify
- SBOM per package, with lifecycle data
- VEX statements
- Support agreement with a service level

STILL YOURS TO PRODUCE

- Source policy naming SecureChain as an approved source
- Intake records for your own decisions
- Your remediation log
- Risk acceptance and removal decisions

TuxCare as your supplier under the CRA, DORA and NIS2

TuxCare is itself a manufacturer under the CRA for SecureChain and for ELS, in the default product category under self-assessment, with a declared five-year support period for the SecureChain catalog. The artifacts you receive are the ones a manufacturer is required to be able to produce, which is why they fit the file in Chapter 9 without translation.

If you are a financial entity under DORA, a SecureChain subscription is a contractual arrangement with an ICT third-party service provider, and it belongs in your register of information. TuxCare provides the details that entry requires.

If you are an essential or important entity under NIS2, your assessment of suppliers' secure development practices can draw on the same artifacts: the SBOMs, the VEX statements, the signatures and provenance attestations, and the declared support period.

TuxCare as a CVE Numbering Authority

Since 14 July 2026, TuxCare has been a CVE Numbering Authority under the MITRE top-level root, authorized by the CVE Program to assign CVE IDs and publish CVE Records for vulnerabilities in products developed, maintained and distributed by TuxCare, including Endless Lifecycle Support and SecureChain.

It closes a gap that runs under Chapters 6 and 7: a vulnerability in a package whose upstream has stopped would otherwise have no one to assign it an identifier, which leaves it untracked in the public databases your scanners, your VEX statements and your remediation log depend on. TuxCare commits to assigning a CVE ID within 24 hours of a triage decision and to publishing the CVE Record within 24 hours of its own advisory, so the identifier, the record and the patch come from the same organization. Vulnerabilities in covered software can be reported to security@tuxcare.com. The standard embargo is 90 days from confirmed triage.

What SecureChain and ELS do not do

They do not write your component policy, run your risk assessment, decide which of your exceptions to accept, or file your regulatory reports. They do not make a migration unnecessary where migration is the right answer; they keep the component patched while you get there. And they cover the open source packages in your dependency tree; your own code is outside their scope.

APPENDIX A

The regulatory timeline, 2024 to 2027

17 Oct 2024 NIS2	●	NIS2 transposition deadline; the implementing regulation for the digital entities it covers adopted the same day.
10 Dec 2024 CRA	●	The Cyber Resilience Act enters into force.
17 Jan 2025 DORA	●	DORA applies to financial entities.
31 Mar 2025 PCI DSS	●	All future-dated requirements of PCI DSS version 4 become mandatory, including the third-party component inventory.
10 Jun 2026 FEDRAMP	●	CISA Binding Operational Directive 26-04 reprioritizes federal vulnerability remediation.
11 Jun 2026 CRA	●	The CRA's framework for notifying conformity assessment bodies becomes operational.
4 Jul 2026 FEDRAMP	●	FedRAMP's 2026 vulnerability detection and response rules become available for adoption.
27 Jul 2026 CRA	●	The European Commission publishes its CRA guidance, C(2026) 5252.
29 Jul 2026 SBOM	●	CISA publishes version 2.1 of its Minimum Elements for an SBOM, co-authored with seventeen partner agencies.
11 Sep 2026 CRA	●	CRA reporting obligations for actively exploited vulnerabilities and severe incidents take effect, including for products already on the market.
7 Dec 2026 FEDRAMP	○	FedRAMP's 2026 vulnerability rules become mandatory for all authorized services.
7 Mar 2027 FEDRAMP	○	FedRAMP grace period ends.
11 Dec 2027 CRA	○	The CRA applies in full. Products with digital elements placed on the EU market must meet the essential requirements and carry CE marking.

● In force as of October 2026 ○ Coming next

APPENDIX B

Glossary

SBOM. Software bill of materials. A machine-readable list of the components in a piece of software, with versions and, in fuller forms, producers, licenses and hashes. CycloneDX and SPDX are the accepted formats.

Top-level and transitive dependencies. Top-level dependencies are the packages you chose directly. Transitive dependencies are the packages those packages pulled in. Most vulnerabilities are found in the second group.

VEX. Vulnerability Exploitability eXchange. A machine-readable statement that a particular vulnerability in a listed component does or does not affect a particular product, and why.

Provenance attestation. A signed record of how a software artifact was built: from which source, by which system, under what conditions.

SLSA. Supply-chain Levels for Software Artifacts. A framework defining levels of assurance for build provenance. Higher levels mean stronger, harder-to-forge claims about how an artifact was produced.

CVD. Coordinated vulnerability disclosure. The process by which a vulnerability is reported privately to the party able to fix it, fixed, and then disclosed publicly.

KEV. CISA's Known Exploited Vulnerabilities catalog, a list of vulnerabilities confirmed to be exploited in the wild, with federal remediation deadlines.

Support period. Under the CRA, the period a manufacturer declares during which it will handle vulnerabilities in a product. At least five years unless the product's expected use is shorter.

Legacy ICT system. Under DORA, a system that has reached end of life, cannot be upgraded or fixed, or is no longer supported by its supplier or by a third-party provider, but is still in use.

ICT third-party service provider. Under DORA, "an undertaking providing ICT services." DORA's third-party rules attach to the contractual arrangements a financial entity holds for those services.

Open source software steward. Under the CRA, a legal entity that provides sustained support for open source software intended for commercial use without acting as its manufacturer. Stewards carry lighter obligations than manufacturers.

Essential and important entities. The two categories of organization covered by NIS2, distinguished by sector and size, with different penalty ceilings.

ICT. Information and communication technology. The EU regulations' umbrella term for IT systems, networks, software and the services around them.

3PAO. Third-party assessment organization. An assessor accredited to perform FedRAMP security assessments of cloud services.

QSA. Qualified Security Assessor. A company qualified by the PCI Security Standards Council to assess compliance with PCI DSS.

CNA. CVE Numbering Authority. An organization authorized by the CVE Program to assign CVE IDs and publish CVE Records for vulnerabilities within its scope.

Notified body. An independent conformity assessment body designated by an EU member state. Required for some CRA product categories; not required for default-category products, which self-assess.

APPENDIX C

Primary texts

Regulation (EU) 2024/2847, the Cyber Resilience Act. https://eur-lex.europa.eu/eli/reg/2024/2847/oj

European Commission, Commission guidance on the application of Regulation (EU) 2024/2847 (Cyber Resilience Act), C(2026) 5252 final, Brussels, 27 July 2026 (Communication and Annex). https://digital-strategy.ec.europa.eu/en/library/commission-publishes-new-guidance-support-timely-cyber-resilience-act-implementation

NIST SP 800-53 Revision 5, Security and Privacy Controls for Information Systems and Organizations. csrc.nist.gov/pubs/sp/800/53/r5/upd1/final. FedRAMP Rev. 5 baselines. fedramp.gov

FedRAMP 2026 Consolidated Rules, Vulnerability Detection and Response; Vulnerability Evaluation and Reporting; Incident Evaluation and Communication. fedramp.gov/2026/reference/20x/ (20x classes) and fedramp.gov/2026/reference/rev5/ (Rev. 5 classes) and fedramp.gov/2026/reference/incident-evaluation-and-communication/

FedRAMP Notice 0014, Response to CISA BOD 26-04, 16 June 2026. fedramp.gov/notices/0014/

CISA Binding Operational Directive 26-04, Prioritizing Security Updates Based on Risk, 10 June 2026. cisa.gov/news-events/directives/bod-26-04-prioritizing-security-updates-based-risk

CISA, 2026 Minimum Elements for a Software Bill of Materials, version 2.1, 29 July 2026. cisa.gov/sbom

PCI Security Standards Council, PCI DSS version 4.0.1, June 2024. pcisecuritystandards.org/document_library/

Directive (EU) 2022/2555, the NIS2 Directive. eur-lex.europa.eu/eli/dir/2022/2555/oj

Commission Implementing Regulation (EU) 2024/2690, technical and methodological requirements under NIS2 for entities in the digital infrastructure, ICT service management and digital provider sectors. eur-lex.europa.eu/eli/reg_impl/2024/2690/oj

ENISA, Technical Implementation Guidance on cybersecurity risk management measures, version 1.0, June 2025. enisa.europa.eu (PDF)

Regulation (EU) 2022/2554, the Digital Operational Resilience Act. eur-lex.europa.eu/eli/reg/2022/2554/oj

Commission Delegated Regulation (EU) 2024/1774, regulatory technical standards on ICT risk management under DORA. eur-lex.europa.eu/eli/reg_del/2024/1774/oj

Commission Delegated Regulation (EU) 2025/301, regulatory technical standards on the reporting of major ICT-related incidents under DORA. eur-lex.europa.eu/eli/reg_del/2025/301/oj

SLSA, Supply-chain Levels for Software Artifacts specification. slsa.dev

CISA, Vulnerability Exploitability eXchange (VEX) documents. cisa.gov/sbom

TuxCare, Vulnerability Reporting and CNA scope. tuxcare.com/vulnerability-reporting/; CVE Program, TuxCare Added as CNA, 14 July 2026. cve.org; TuxCare, ELS for Language Ecosystems machine-readable security data. docs.tuxcare.com

TuxCare

We Secure Open Source

www.TuxCare.com

