Move2Cloud

• WHITE PAPER — 2027 EDITION

DEVSECOPS & APPLICATION SECURITY

Software Supply Chain
The New Cybersecurity Front Line

SBOM, CI/CD pipeline security, code signing and continuous open source dependency management — how to structure a DevSecOps approach in the face of a fast-growing threat and a European regulatory deadline.

11 pages · CI/CD controls comparison, self-assessment grid and roadmap included

30%
of breaches involved a third party in 2025, up from 15% a year earlier (Verizon DBIR)

$4.91M
average cost of a software supply chain breach

+75%
more new malicious open source packages detected in one year (Sonatype)

PREPARED BY
Move2Cloud — Cloud & Infrastructure Consulting, Paris

PUBLICATION
September 2026

CONTENTS

What you will find in this document

A six-step journey, from understanding the threat to an operational DevSecOps roadmap.

01 **Executive summary — the essentials on one page** p.3

02 **Why the software supply chain has become target no.1** p.4

03 **SBOM: from best practice to regulatory obligation** p.5

04 **Shift-left: embedding security in the CI/CD pipeline** p.6

05 **Signature and provenance: guaranteeing the integrity of deployed code** p.7

06 **Managing open source dependencies continuously** p.8

07 **Self-assessment grid** p.9

08 **A 5-step roadmap** p.10

09 **Conclusion & about Move2Cloud** p.11

> **Who is this document for?** CIOs, CISOs, platform and development teams whose applications depend — as almost all modern applications do — on open source components and a CI/CD pipeline.

Copyright Move2Cloud — move2cloud.fr

EXECUTIVE SUMMARY

The essentials on one page

Application security is no longer played out solely in the code we write — it is played out just as much in the code we assemble, often without knowing precisely what it contains.

According to Verizon's 2025 Data Breach Investigations Report, **30% of breaches involved a third party** — a doubling compared with 15% the previous year, described as the single largest year-on-year shift in the report's history. A software supply chain compromise costs on average **4.91 million dollars**, around 11% more than the overall average for incidents, and takes on average **267 days** to identify and contain — the longest detection time of all the attack vectors tracked.

This pressure also comes from sheer volume: according to Sonatype, more than **454,600 new malicious open source packages** were identified in 2025 across the npm, PyPI, Maven, NuGet and Hugging Face registries — a 75% increase in one year, with more than 99% of that volume concentrated on the npm registry alone. Against this backdrop, the European regulatory framework is tightening: since September 2026 the Cyber Resilience Act has already imposed a vulnerability notification obligation, with full compliance — SBOM, technical documentation, CE marking — required by December 2027.

For any organisation that publishes or operates software — which, with applications made up on average of more than 80% open source code, means practically every organisation — the question is no longer "are we using third-party dependencies?" but "do we know precisely which ones, and how they are secured all along the pipeline?". This document offers a framework for answering it.

30%
of breaches involve a third party in 2025, versus 15% a year earlier

$4.91M
average cost of a software supply chain compromise

267 d.
average detection time — the longest of all attack vectors

For any organisation that publishes or operates software — which, with applications made up on average of more than 80% open source code, means practically every organisation — the question is no longer "are we using third-party dependencies?" but "do we know precisely which ones, and how they are secured all along the pipeline?". This document offers a framework for answering it.

SECTION 02

Why the software supply chain has become target no.1

A modern application is, for the most part, no longer code written in-house: it is an assembly of open source dependencies, base containers, CI/CD tooling and third-party services. This composition, which has multiplied development speed, has also multiplied the attack surface — and shifted the attackers' target.

1. Compromise one component and you compromise all its users

An attacker who manages to inject malicious code into a widely used open source package no longer needs to target each organisation individually: the compromise propagates automatically to every build that includes that package. It is this propagation mechanism that makes supply chain attacks disproportionately effective relative to their cost for the attacker.

2. Detection remains structurally late

An average of 267 days to identify a supply chain compromise means, in practice, that the malicious code has had time to be deployed to production, potentially several times over, before it is discovered. This latency stems from the difficulty of manually auditing thousands of transitive dependencies.

3. Open source registries have themselves become direct targets

The 75% growth in malicious packages detected in a single year does not only reflect more opportunistic attacks: it reflects an industrialisation of the publication of typosquatted or compromised packages, with a very marked concentration on the most widely used registries such as npm.

“ The question is no longer whether a component of your supply chain will be targeted — it is whether you will detect it before it reaches production. ”

The following sections set out, control by control, how to structure a DevSecOps approach that answers that question.

Copyright Move2Cloud — move2cloud.fr

SECTION 03

SBOM: from best practice to regulatory obligation

A SBOM (Software Bill of Materials) is the structured, machine-readable inventory of every component that goes into a piece of software — the prerequisite for any serious management of supply chain risk.

DEADLINE	WHAT IT REQUIRES
11 September 2026	Vulnerability and incident notification obligations come into force for manufacturers of products with digital elements, under the European Cyber Resilience Act (CRA).
11 December 2027	Full compliance required: machine-readable SBOM covering at minimum first-level dependencies, technical documentation, risk assessment and CE marking.
Today	Without an up-to-date SBOM, an organisation generally discovers it is exposed to a critical vulnerability the same way as the general public — a media alert followed by an urgent manual search — whereas with a continuously maintained SBOM the same question is answered by a simple query.

> **Key takeaway:** The CRA applies to any company that places a digital product on the European market, wherever its head office is located — French software vendors cannot consider themselves out of scope simply because of where they are based.

Beyond compliance: an incident response tool

The SBOM is not merely a documentation obligation: when a critical vulnerability is disclosed in a widely used library, having an up-to-date SBOM makes it possible to answer the question that really matters — are we exposed, and where? — in hours rather than weeks.

Copyright Move2Cloud — move2cloud.fr

SECTION 04

Shift-left: embedding security in the CI/CD pipeline

The "shift-left" principle consists of moving security controls as early as possible in the development cycle — where a fix costs minutes, not days of lost production.

PIPELINE STAGE	CONTROL TO INTEGRATE	HOW IT WORKS	WHAT IT CATCHES
Commit / pull request	SAST and secret detection	Run automatically before any merge into the main branch.	Code-level flaws and hard-coded secrets.
Build	SBOM and SCA	Automatic SBOM generation and software composition analysis at each build.	Vulnerable or obsolete dependencies.
Containerisation	Container image scanning	Images scanned before publication to a registry, with automatic blocking.	Images containing known critical vulnerabilities.

> **Common pitfall:** A pipeline that systematically blocks on every alert, including minor ones, quickly pushes teams to work around the controls rather than comply with them.

Failing the build early rather than auditing at the end

The goal of shift-left is not to add further controls at the end of the cycle, but to fail a build as soon as a problem is detected — the cost of fixing a vulnerability rises sharply with every stage it passes through undetected. The chain ends at deployment, where the signature and provenance of artefacts are verified before anything reaches production (detailed in section 05), and it holds over time only with a graduated severity policy: automatic blocking on critical findings, non-blocking alerts on minor ones.

SECTION 05

Signature and provenance: guaranteeing the integrity of deployed code

Knowing that a component is clean when it is integrated is not enough — you also need to guarantee that the artefact actually deployed in production is the one that was validated, with no intermediate alteration.

Sign every artefact produced by the pipeline

Every container image, package and binary produced by the CI/CD pipeline should be cryptographically signed at build time, with systematic verification of that signature before any deployment.

Trace provenance, not just integrity

Beyond signing, a provenance attestation documents exactly how an artefact was built: from which source code, with which tools, on which infrastructure — revealing an abnormal build before its content is even analysed.

Verification at deployment

This practice prevents an artefact modified after the fact — whether through a compromise of the registry or of the deployment infrastructure — from being executed without detection.

CONTROL	WHAT IT GUARANTEES
Cryptographic signature	The deployed artefact is identical to the one produced and validated by the pipeline, with no intermediate alteration.
Provenance attestation	The artefact does come from the official pipeline and the expected source code, not from a parallel or compromised source.

Copyright Move2Cloud — move2cloud.fr

SECTION 06

Managing open source dependencies continuously

A point-in-time dependency inventory is out of date the moment it is published — managing open source dependencies only has value if it is continuous.

Update, don't just detect

Detecting a vulnerable dependency without an established process to update it simply moves the problem to a dashboard nobody reads. Automating minor dependency updates and security patches — with automated tests confirming the absence of regressions — remains the most reliable way to keep an application estate up to date without tying up a dedicated team on an ongoing basis.

Assess the project's health, not just its popularity

A package's download count says nothing about how well it is actually maintained. More reliable signals: how frequently security patches are published, the number of active maintainers, and a track record of responding quickly to reported vulnerabilities — criteria to apply when a dependency is first chosen, not only during a retrospective audit.

> **Key takeaway:** No control taken in isolation — scanning, signing, automated updates — is enough to eliminate supply chain risk. It is their combination, applied continuously rather than as a point-in-time audit, that genuinely reduces exposure.

Beyond prevention, limiting the permissions granted to build and deployment processes — restricted network access, short-lived authentication tokens with minimal scope — reduces the impact of a malicious dependency that slips past the earlier controls, without preventing the incident entirely.

Copyright Move2Cloud — move2cloud.fr

SECTION 07

Self-assessment grid

Before considering your software supply chain under control, check these seven controls.

- [ ] **Is there an up-to-date SBOM for each of your critical applications?** Without one, answering "are we exposed?" during a security alert takes days rather than hours.
- [ ] **Is software composition analysis (SCA) run automatically on every build?** A one-off analysis will not detect a dependency that became vulnerable after it was integrated.
- [ ] **Is every artefact deployed to production signed, and its signature verified before execution?** Without this, nothing guarantees that the artefact deployed is the one that was validated.
- [ ] **Are security updates to dependencies automated, or do they depend on a periodic manual review?** A manual review systematically falls behind the volume of packages to keep up to date.
- [ ] **Do the access tokens used by the CI/CD pipeline have a limited scope and lifetime?** An over-permissive token turns a minor pipeline compromise into extended access.
- [ ] **Does your organisation know whether it falls within the scope of the Cyber Resilience Act?** Any vendor marketing a digital product in the EU may be concerned, regardless of its size.

> **How to read your answers:** Three "no" answers or more indicate that a structured audit of the software supply chain should precede any further extension of the exposed application estate.

Copyright Move2Cloud — move2cloud.fr

SECTION 08

A 5-step roadmap

A progressive approach, designed to embed supply chain security without slowing down the teams' delivery pace.

1 **Take inventory**

Generate a first SBOM for each critical application, in order to establish precisely the real composition of the application estate.

2 **Automate detection**

Integrate software composition analysis (SCA) and container scanning directly into the CI/CD pipeline, with a graduated severity policy.

3 **Sign and verify**

Put in place systematic signing of artefacts and blocking at deployment of any unverified artefact.

4 **Automate updates**

Deploy an automated dependency update tool, coupled with automated tests guaranteeing the absence of regressions.

5 **Prepare for CRA compliance**

Formalise the generation and continuous maintenance of the SBOM, and document the vulnerability management process ahead of the December 2027 deadline.

Copyright Move2Cloud — move2cloud.fr

CONCLUSION

Supply chain security, a continuous undertaking

With 30% of breaches now involving a third party and a tightening European regulatory framework, software supply chain security is no longer a topic reserved for large organisations with dedicated security teams.

Organisations that embed SBOM, continuous scanning, signing and automated updates directly into their pipeline — rather than as a one-off audit — reduce both their actual exposure and their response time on the day an incident occurs despite everything.

Move2Cloud	is a consulting firm specialised in infrastructure, cloud and IT transformation. We support organisations in the audit, the securing and compliance of their cloud environment.
Contact	Move2Cloud — Paris · move2cloud.fr
Next step	A DevSecOps audit makes it possible to answer the seven questions on page 9 within a few weeks, with a prioritised remediation plan.

Sources

Verizon, "2025 Data Breach Investigations Report" (summary via swif.ai, "Supply Chain Attack Statistics for 2026") — Sonatype, "2026 State of the Software Supply Chain Report" — Mend.io, "EU Cyber Resilience Act: 2026 Compliance Guide" — Regulation (EU) 2024/2847 (Cyber Resilience Act)

Document written by Move2Cloud as an example and illustration, for educational purposes. The regulatory deadlines presented reflect the state of the European calendar at the publication date and do not constitute legal advice.