Rezilion

# The Role of the SBOM in Securing the Software Supply Chain

## Introduction

**A SOFTWARE SUPPLY CHAIN COMPRISES ALL OF THE COMPONENTS,** libraries, tools, and processes used to develop and build an application throughout the entire Software Development Life Cycle (SDLC). Software vendors often create their products by gathering open source and commercial software components.

The number of software users increases daily, and as such, so does the opportunity for vulnerabilities to be introduced. Organizations should monitor companies throughout the supply chain and the open source code that many developers utilize. However, extensive use of code does not necessarily mean there is sufficient vulnerability scrutiny.

Because it is so critical to the software development process, the software chain has become a target and is under constant attack with high-profile breaches such as the ones impacting SolarWinds and Kaseya.

What exactly does this mean? A software supply chain attack occurs when software at any stage of the development, delivery, and usage cycle is compromised and an attacker gains unauthorized access to development environments and infrastructure. This can include *version control systems, artifact registries, open source repositories, continuous integration pipelines, build servers, or application servers. Once they gain access to an organization's environment, an attacker can modify source code, scripts, and packages, and establish back doors to steal data from within.*

This is a problem that is only expected to grow worse. In fact, a recent report by Gartner finds that by 2025, 45% of organizations worldwide will have experienced attacks on their software supply chains — a three-fold increase from 2021.

Compound that with the fact that 17% of breaches at critical infrastructure organizations were caused due to a business partner being initially compromised, highlighting the security risks that come from trusting third-parties are doing their due diligence.

In 2021, the Biden Administration issued a cybersecurity executive order that highlights the importance of adopting a zero trust approach to strengthen the nation's cybersecurity. Yet, only 21% of critical infrastructure organizations studied have adopted a zero trust security model, according to IBM's annual 2022 Cost of a Data Breach report.

"Too much of our software, including critical software, is shipped with significant vulnerabilities that our adversaries exploit," the order observed. "This is a long-standing, well-known problem, but for too long we have kicked the can down the road. We need to use the purchasing power of the federal government to drive the market to build security into all software from the ground up."

The federal government set the standard, but the private sector must do its part as well. This white paper will illustrate how organizations can ensure the safety of the software supply chain and protect it from breaches.

## When Breaches Occur in the Software Supply Chain

**LOG4SHELL, A SOFTWARE VULNERABILITY IN APACHE LOG4J** (a Java library for logging error messages in applications), sent shockwaves through the industry after it enabled massive exploitation of victims' devices for cryptocurrency mining, and ransomware attacks, sending spam, establishing backdoors, and other activities. It continues to resonate today.

Meanwhile, software supply chain attacks are becoming increasingly sophisticated, with malicious actors exploiting weaknesses at every stage of the software procurement, development, and delivery life cycle. Techniques include everything from injecting malicious code into open source packages to installing back doors in post-deployment software updates.

While security is crucial for all organizations, enterprises in regulated industries, such as financial services, healthcare, and connected devices face a more daunting challenge. Achieving security and proving compliance while improving and innovating their products has proven to be challenging. Companies developing software for connected devices and SaaS companies have strict requirements for security — but cannot spend all their time on patching and neglect development.

Open source software is a core part of modern software development. However, there is a lack of visibility and transparency into proprietary and open source dependencies within the software supply chain, which exacerbates security and compliance risks, according to Gartner.

"Software engineering teams often lack the tools, practices, and standards to systematically discover and share details about vulnerable software packages across the organization," the firm observed.

Software supply chain security attacks have exposed the risks associated with commercially procured tools and platforms because you don't know what's "inside the box," Gartner said.

The nature of software is that it changes and evolves over time due to optimization, new features, and security fixes. As a result, developers throughout the supply chain have to continually evaluate how changes might impact their software. This includes changes to third-party components used to compose software.

When there is a breach in the supply chain, product delivery can be impacted. Once dependencies are understood, there needs to be visibility into the software supply chain.

## The Evolution of SBOMs to Dynamic SBOMs

**SBOMs ARE DESIGNED TO ENHANCE VISIBILITY INTO THE COMPLETE SOFTWARE SUPPLY CHAIN** by tracking dependencies between open source components in the SDLC as well as ensuring compliance with open source licenses, provenance, and known vulnerabilities.

That enhanced visibility into the codebase leads to better prioritization and quicker delivery of code updates.

One of the greatest value-adds for SBOMs is the ability for end users to monitor vulnerabilities in parallel with whatever vulnerability management is conducted by the supplier, according to The National Telecommunications and Information Administration (NTIA). The ability to "trust but verify" and continuously monitor the vulnerability status of a supplier's software dependencies provides continuous assurance by eliminating gaps in situational awareness.

For example, it reduces the time it takes to detect vulnerabilities, remediate them, and for a supplier to ship a remediated update.

Keeping SBOMs data in sync with corresponding software artifacts presents a key challenge, however. And having visibility isn't enough — you also need to be able to identify any new software vulnerabilities. To do so requires having a Dynamic Software Bill of Materials (SBOM) SBOM, which provides an inventory of all components as the starting point, and then the ability to update it as software is added.

This is helpful to a software engineering team because a Dynamic SBOM responds to real-time changes in the software environment to create and maintain an accurate software supply chain. This capability is crucial for organizations that build and manage many software products.

During the build phase, the Dynamic SBOM supports testing for vulnerabilities while integrating the changes to code. Upon release, the Dynamic SBOM continues to reflect any software changes. When using a traditional SBOM, concurrent versions often occur with no reconciliation, which leads to missing components and complex version management. With patch deployment, the SBOM must reflect the changes.

A Dynamic SBOM shows the organization's software components and provides accurate and real-time information — throughout the SDLC. Using a Dynamic SBOM provides visibility and enables security control management and implementation across the attack surface, from development to production.

**Using a Dynamic SBOM provides visibility and enables security control management and implementation across the attack surface, from development to production.**

## Several Business Units Can Benefit from Dynamic SBOMs

**ONCE YOU HAVE A DYNAMIC SBOM,** it's time to leverage it and share it with different groups of people in the organization who require it. This includes not only security professionals, but those in the compliance, procurement, and legal business units as well. All of these personas will have different software requirements to comply with the supply chain. Each one needs visibility into the supply chain for their specific job roles. For example:

a. **Software procurement —** A software buyer for the federal government needs a Dynamic SBOM because without that they can't make purchases. This helps them determine what standards they should use.

b. **Legal department —** Lawyers must ensure all compliance and end-of-life requirements for components are in place because those software components will have to be replaced.

c. **Developers —** When developers are writing code, they must have a central repository of the approved components they can use that are secure.

d. **Vendors —** A supplier's Dynamic SBOM must detail third-party software dependencies that are both open source and proprietary and incorporate them into a product, as well as installed dependencies required at runtime, according to the NTIA. This will provide insight into vulnerability management and asset management and will help vendors manage licensing and compliance and quickly identify software or component dependencies and supply chain risks.

   Further, if a supplier product installs a third-party dependency on the consumer's system, either as a constituent component of a product or as an installed enabling capability, it should be enumerated in a Dynamic SBOM to provide transparency for software asset management and vulnerability management. Failing to list installed runtime dependencies leaves a dangerous gap in situational awareness that exposes the consumer to compromise when an outdated and vulnerable installed dependency is exposed and a remediated update has not been provided by the supplier.

e. **Product security —** Security teams need to make sure that there are no significant changes in the software product from development to release by tracking all their components.

## React with Automation

**WITH BETTER, CONTINUALLY UPDATED INFORMATION,** organizations can filter the noise and prioritize the most dangerous vulnerabilities or the ones that affect the most applications. By prioritizing the threats and dealing with the most impactful ones first, developers can focus on vulnerabilities their organization's policy defines as exceeding their risk tolerance. Ultimately, this practice reduces patching efforts and leaves developers more time to innovate.

Without automation, visibility and prioritization are not feasible. An automated and Dynamic SBOM is integral to providing real-time component visibility. Further, the means of scanning and filtering vulnerabilities must be automated to reduce remediation time and enable companies to benefit from this approach. Using dynamic tools to monitor vulnerabilities helps organizations catch and remediate the most essential vulnerabilities before exploitation. This addresses the recent concerns with supply chain vulnerability management, allowing organizations to use third-party or open source code while negating the risk.

The ideal process should employ scanning automation and filter vulnerabilities based on exploitability and policy definition and then remediate the vulnerabilities, starting with the most detrimental. This process ensures greater time efficiency, less inappropriate action, and a better relationship between DevOps and security teams.

Dynamic SBOMs are invaluable for gaining visibility into how software gets built, the components that make up that software, and for improved data sharing. They also help identify security vulnerabilities more quickly so they can be remediated.

When teams discover flaws or vulnerabilities in a component, they can use Dynamic SBOMs to quickly identify all software that is affected by the vulnerable component. SBOMs also provide them with information to assess the potential impact and risks introduced by the vulnerable component. From there, they help determine whether those vulnerable components are developed internally, commercially or from open source libraries.

Through Dynamic SBOM and automation, businesses can improve efficiency and outcomes from development to production. It costs less to reach feature functionality when developers spend less time on patching and more on coding. Without having to spend as much time patching, developers become more productive and can innovate faster and more securely.

The bottom line? In today's software development climate, you cannot adequately protect the software supply chain without a Dynamic SBOM.

## Secure Your Supply Chain with a Dynamic SBOM Today

**WITH REZILON'S DYNAMIC SBOM,** customers know their real attack surface as it changes dynamically. The platform seamlessly plugs into all software environments, from development to production, and provides full-stack coverage of third-party and home-grown software across hosts, containers, and application layers.

Unlike static SBOMs, Rezilion's Dynamic SBOM does more than just uncover what software components are there: It reveals if and where they're being executed in runtime (if loaded to memory, they are exploitable, if not loaded, they don't pose a risk), providing organizations with an unparalleled solution to understand where bugs exist — but also whether they could be exploited by attackers.

Rezilion makes it easier for teams to manage and eliminate software vulnerabilities.

- ✓ Inventory all of your software components in real time with a Dynamic Software Bill of Materials (SBOM).
- ✓ Pinpoint specific vulnerabilities and know if they're exploitable in your environment.
- ✓ Filter out the noise by eliminating up to 85% of vulnerabilities that don't pose any risk.

Get started on a new path to managing software vulnerabilities and book a demo to see our Dynamic SBOM in action today.

**For more information,** visit www.rezilion.com/platform/dynamic-sbom/ and see it in action and book a demo at www.rezilion.com/lp/dynamic-sbom-book-a-demo/.

**About Rezilion**

Rezilion's platform automatically secures the software you deliver to customers. Rezilion's continuous runtime analysis detects vulnerable software components on any layer of the software stack and determines their exploitability, filtering out up to 95% of identified vulnerabilities. Rezilion then automatically mitigates exploitable vulnerabilities across the SDLC, reducing vulnerability backlogs and remediation timelines from months to hours, while giving DevOps teams time back to build.

Learn more about Rezilion's software attack surface management platform at www.rezilion.com and get your 30-day free trial.