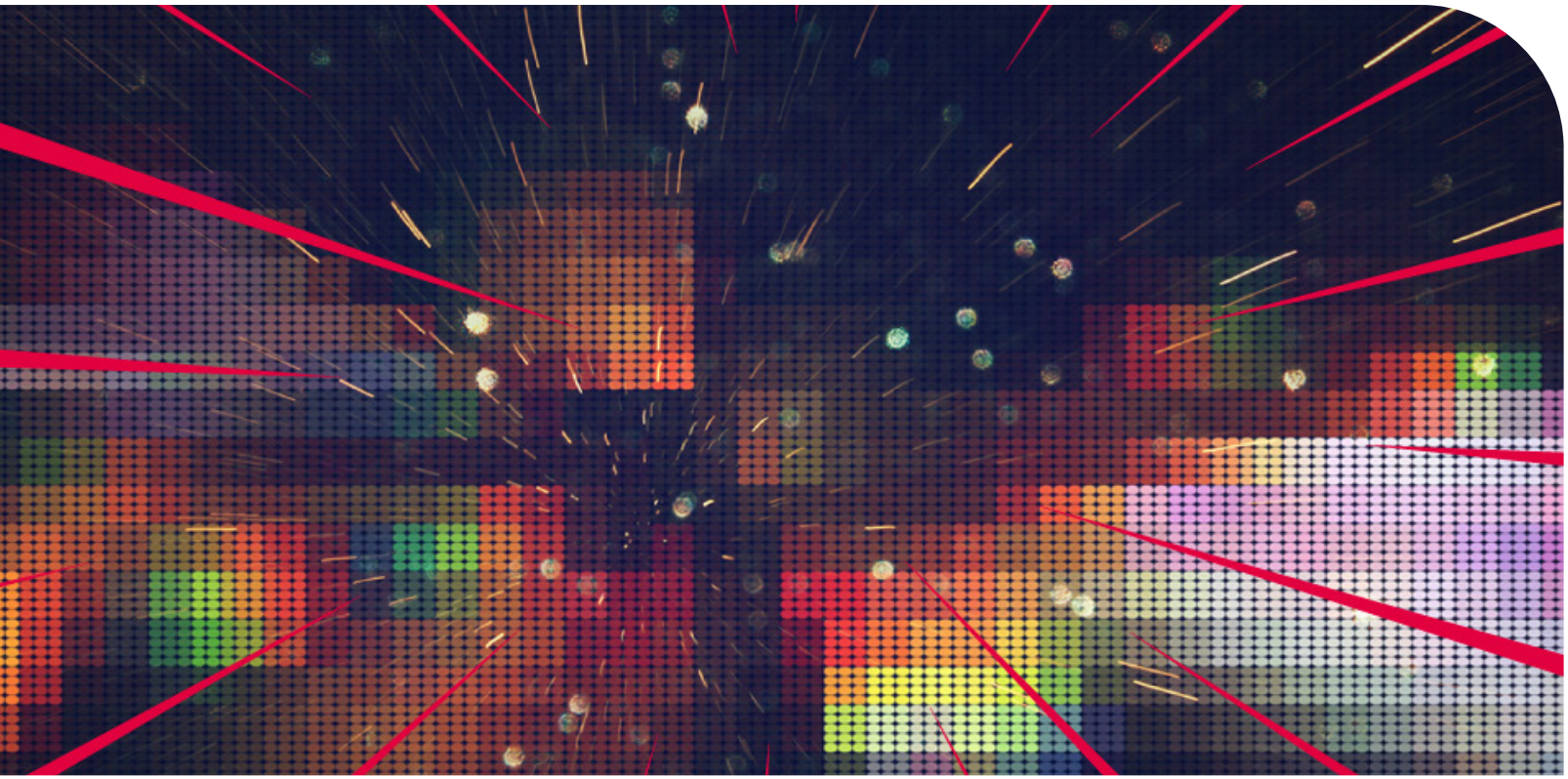
Rezilion

EBOOK

Why SBOMs need to evolve into Dynamic SBOMs



It's not like a Bill of Materials (BOM) is a new concept. It has been around for decades and used extensively in the manufacturing industry to track, resolve, and manage production issues. An SBOM, however, is a fairly new application of BOMs.

SBOMs or Software Bill of Materials have been in the news quite frequently in the past 12 months, due mostly to the SolarWinds and Kaseya supply chain cyberattacks and Executive Order 14028. In addition, the Log4j attack really put the SBOM in focus for a lot of organizations, vendors, and users of software.

All of us in cybersecurity need to understand the importance of SBOMs and educate others on the need for one. It's a critical element of the overall cybersecurity discipline.

AS THE WORLD GOES DIGITAL AND MOVES TO THE CLOUD, organizations are producing more software products than ever before. They are facing some daunting security challenges:

- The software attack surface keeps growing through software innovation. Translation: Millions of lines of code, and an equal amount of components, mean elevated risk.
- The components are diverse and from different sources. These components range from images, files, packages, and libraries to open source (OSS) and third-party components. It's a very complex environment to manage.
- The use of OSS and third-party components creates an additional layer of complexity for product organizations as they have to extend security outside of their organizations to ensure that the components used are safe and secure.

- The software attack surface is difficult to manage as it is spread out across a broad technology stack that includes clouds, containers, applications, hosts, IoT devices, etc., at different stages of the DevOps lifecycle from development to production. This lack of perimeter makes it very difficult.
- This growth in the software attack surface has created an opportunity for threat actors to find new ways to attack. Attacks like Solarwinds and vulnerabilities like Log4j, and Spring4Shell are examples of how threat actors have taken advantage of this software attack surface.

What makes this even more difficult is it creates an environment where the teams don't have enough visibility into their software attack surface. Vulnerabilities keep growing, and security teams are unable to catch up. It hinders their ability to develop and release products quickly and securely everyday.

To mitigate software security threats, security leaders must:

- Truly understand their software attack surface at any given time.
- Know which components of the software attack surface are vulnerable and exploitable.
- Understand the supply chain risk associated with their components.
- Have the critical information needed to prioritize software risks.
- Be positioned to mitigate and remediate risks quickly.

As organizations grow and innovate, they increase the diversity of the components used, their source, and their use. With little to no understanding of their true software attack surface, its composition, and exploitability in real time, it is not possible for developers and product security teams to identify, validate, prioritize, and remediate their enterprise and supply chain risk effectively.

Enter the SBOM.

SO, WHAT IS AN SBOM? Simply put, it is an inventory of all the elements or components that make up software. These components range from packages, files, libraries, and open source software, to third-party components used in the development of software. With an SBOM, an organization can get an understanding of what components are present in their software/applications, know how the components are related (dependencies), identify vulnerabilities, and understand the risk associated with them managing both their enterprise and supply chain risk.

Generally an SBOM contains the following data fields:

1. Author/Supplier
2. List of components and their names
3. Component relationships and dependencies
4. Component hash
5. Version
6. Licensing information

The basic characteristics of SBOMs include the following:

- SBOMs must be machine-readable and have the ability to be generated automatically.
- SBOMs should have the minimum set of data fields as outlined above.
- SBOMs should be exportable and standardized. Currently there are three main formats that are in use:
  - CycloneDX (https://cyclonedx.org/) is a lightweight SBOM export format that is driven by the OWASP foundation. It incorporates elements of SPDX and SWID.
  - SPDX (https://spdx.dev/) is a standard that is being driven by the Linux foundation. It is also known as ISO/IEC 5962:2021. SPDX formatted SBOMs provide security, provenance, and licensing details regarding the software.
  - SWID, this format is available in four tags — corpus tags, primary tags, patch tags, and supplement tags which are used to identify and report the different software components. Also known as ISO/IEC 19770-2:2015.
- SBOMs must be auditable.
- SBOMs must be shareable, exportable, and integratable for broader adoption and use.

NOTE: No single format is emerging as the dominant one, vendors and users can decide which format is better for their use cases. According to NTIA "Due to the diverse needs of the software ecosystem, there is no one-size-fits-all solution; however, modeling SBOM processes on existing approaches and methods will enable interoperability between vendors, dampen variance, minimize the need for new tools, and thereby simplify processes required for better supply chain management."

SBOMs PROVIDE CRITICAL VISIBILITY INTO THE SOFTWARE ECOSYSTEM OF AN ORGANIZATION and are vital to understanding its composition. SBOMs provide clear insights into the enterprise and supply chain risks. Key use cases for an SBOM include:

- Discovering all software components.
- Understanding which components are vulnerable and which are not.
- Understanding software supply chain risk, including provenance and license compliance.
- Compliance with your organization's procurement and software transparency requirements.
- Meeting EO 14028 requirements.
- Understanding and mapping the journey of your software components from development to production.
- Prioritizing risk and managing remediation.

As such, they are useful and important to a broad set of stakeholders:

- Security and development teams:
  - Various stakeholders such as developers, application security engineers, product security teams, vulnerability management manager, cloud, and infrastructure security teams.
- Security analysts involved in threat investigation and threat response.
- Procurement teams responsible for acquiring software and ensuring compliance to organizational standards.
  - Vendor risk management
  - Legal/compliance mandate
  - Supply chain risk
- CISOs for oversight and reporting.

In a recent study published by the Linux foundation Software Bill of Materials (SBOM) and Cybersecurity Readiness, they noted the top 3 benefits from generating an SBOM:

- 51% stated that it is easier for developers to understand dependencies across the software components.
- 49% believed that it was easier to monitor vulnerabilities of the software components with an SBOM.
- 44% thought SBOMs made it easier to manage license compliance for open source software.

It is also very interesting to note that:

- 47% of organizations use SBOMs today.
- 66% of organizations will use an SBOM in 2022.
- 88% of organizations will use an SBOM by 2023.

Clearly there is a growing trend towards the use of SBOMs in order to get a better understanding of the supply chain risks.

IN SPITE OF ALL OF THE DIFFERENT BENEFITS OFFERED BY SBOMS, they are not perfect, and have some challenges.

- **SBOMs are static,** point in time documents that are not updated. Software is dynamic. So, whenever there is a change in the software, a new SBOM has to be created. If SBOMs cannot be easily updated, it results in an ever increasing number of SBOMs. This also adds to the maintenance challenge and their value is greatly diminished.
- **SBOMs are not fully automated,** and without automation, new SBOMs have to be manually created. It is easy to miss risks that may have been caused by software updates. SBOMs are more effective as a security tool if their management is automated.
- **SBOMs are specific to certain software or applications.** Current SBOMs are related to a specific software and/or application/stack and do not provide complete third party components information. This makes their use limited because modern software environments have multiple software/applications in development and production. This requires creating multiple SBOMs, which makes management even more complex.
- **SBOMs require an additional artifact for context.** SBOMs by themselves offer limited value when understanding the exploitability of a specific vulnerability unless they provide the additional context needed to understand if the components and vulnerabilities are actually exploitable. This document is called VEX (Vulnerability eXploitability Exchange) which is a machine readable companion to the SBOM adding to further management complexity.
- **Current SBOM outputs are very large and noisy,** which makes them difficult to understand and adds extra work for the users, developers, product security teams, CISOs, and legal and compliance officers that maintain them.

The current manual and static approach to SBOMs, though valid, does not offer enhanced security of the software supply chain. The way to address these challenges is to make the SBOM dynamic.

SOFTWARE IS NOT STATIC. It changes many times over its development lifecycle. So, if the software is dynamic, then automatically, an effective SBOM should also be dynamic. The following are characteristics of a dynamic SBOM:

- **They are continuous and fully automated:**
  - Automatically updates as your environment changes
  - No need to maintain multiple versions
  - Offers continuous risk assessment
- **They are dynamic:**
  - Provides context on what is exploitable or not exploitable in your environment. At Rezilion, we use runtime analysis to show what is loaded to memory and exploitable
  - Know component state to triage active versus latent threats
- **They are end-to-end:**
  - Provides visibility into your entire software environment from dev to production
  - Covers a broad range of software component types such as containers, applications, hosts, packages, and files.
  - Available and at every stage of the DevOps lifecycle across the entire technology stack
- **They are searchable and customizable:**
  - You can search for any component by every identifier of the component
  - You can search known vulnerabilities to know if any instances exist in your environment
  - You can customize a dynamic SBOM to address your needs
- **They are contextualized:**
  - Component vulnerability context is built in rather than needing a separate VEX document
- **Exportable in multiple formats:**
  - As a standard CycloneDX format

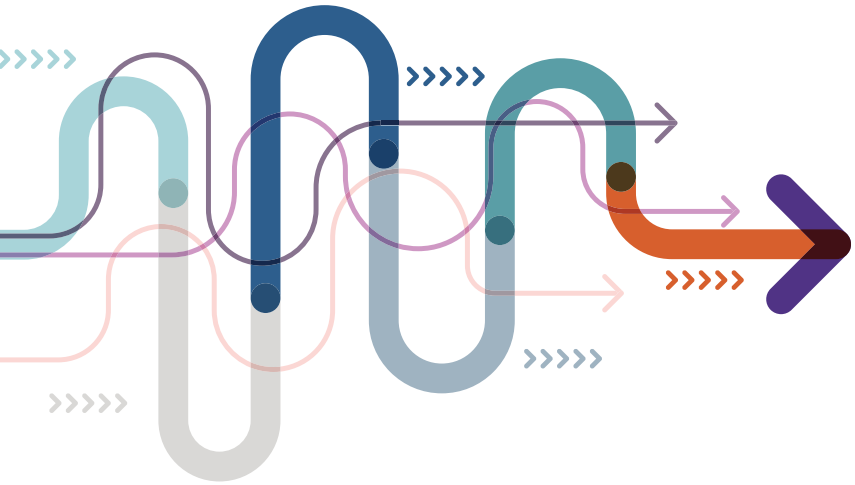
CHAPTER 6: Benefits of the Dynamic SBOM?

- **Dynamic Visibility —** Continuous tracking and management of the software environment and components as changes are being introduced for a true understanding of your software attack surface. Instantly view which components are actually being used and whether they impact your attack surface.
- **Dynamic Identification —** Instantly search and pinpoint vulnerabilities and components (such as Log4j) across billions of files and instantly determine whether or not they are exploitable in your environment. Quickly respond to address any potential risk.
- **Dynamic Context —** Know down to the function level what every component is doing in runtime to triage active versus latent threats.
- **Full Stack, Full Cycle Coverage —** See all software components across CI/CD pipelines and production, on-prem and cloud, hosts, and containers.
- **Share your SBOM with Exportable Formats —** Share important information with customers, auditors, and compliance officers using a formal VEX or Cyclone DX document to facilitate transparency and compliance.
- **Assure Your Customers —** Communicate important vulnerability information with your customers and security teams with built in validation in your dashboard or using a formal VEX document to outline the actual impact of vulnerabilities they may detect in your product.
- **Detect drift in real-time whenever there is change** from stage to another and instantly verify if the change is or isn't authorized. Quickly alert security teams so that proper mitigation or rmediation steps can be taken.
- **Eliminate any coverage gaps** by continually updating the SBOM.
- **Manage your supply chain risk —** Know where everything came from and risks associated with it.
- **Stay 100% prepared** for any software audits and reduce business disruptions during audits.

CHAPTER 7: Key Use Cases of the Dynamic SBOM

WITH A GROWING SOFTWARE ATTACK SURFACE, developers and product security teams must ensure that the products are released and secured quickly. The need to reduce the time to fix the issue is also underscored by the need to shorten the attack window. A dynamic SBOM is one such tool that allows security teams to secure and release quickly. A few of the most important use cases are:

- **Know your software attack surface:** Get a comprehensive end-to-end view of your software attack surface.
- **Understand supply chain risk:** Quickly identify and search specific software components for vulnerabilities within your environment.
- **Understand security risk:** Immediately identify which components of your SBOM are exploitable and which are not using run time analysis.
- **Reduce operational costs** by preventing developers from manually searching for vulnerabilities.
- **Manage operational risk** by ensuring that only acceptable and standardized components are used and vulnerabilities are addressed in a timely manner.
- **Track changes:** Know whenever there is a change and what has changed with an automated dynamic SBOM.
- **Achieve compliance:** Instantly create and share the inventory and documentation necessary to comply with government SBOM requirements including license compliance.
- **Search and analyze** the impact of a specific component (aka Log4j) or specific threats and trigger Dev/Ops to fix them.
- **End of Life (EOL) planning** by looking for alternatives for components that are approaching their EOL.
- **Ensure quality assurance** and product quality by verifying.



JUST LIKE ANY OTHER TECHNOLOGY, NOTHING REMAINS STATIC. Neither will the SBOMs. The future of the SBOM, clearly, must be dynamic if it is to address the different use cases as we have outlined in this ebook. Some of the evolution we've seen in SBOMs are:

- It will take some time for clear standards around SBOMs. Multi-format SBOMs will continue to evolve. Utilities that translate one format to another will offer the interchange flexibility needed. Proprietary SBOMs in non-standard format will continue as stop gaps for businesses where SBOMs are not a mandatory requirement.
- Mega SBOMs combining multiple SBOMs in different formats across different stages of the development lifecycle will be required to get a comprehensive view into the final product.
- Central distribution of SBOMs will create repos and secure distribution platforms to build trust and confidence among the vendors and users.
- SBOM ecosystems will emerge. It will combine multiple enrichment feeds, internal and some external to the organization, to create granular context to establish exploitability.
- Minimum requirements will become more stringent:
  - Machine readability will be a mandatory requirement. Most likely it will be required in SPDX or CycloneDX formats.
  - Dependencies will be expected in all SBOMs.
  - Updated SBOMs will be needed whenever there is change in code or application.
- New use cases around supply chain security will emerge. These include:
  - Supply chain repo SBOMs outlining verified open source components available to developers.
  - Supply chain resilience will drive automated updates on supply chain partners and any security risks associated with their components.
  - Supply chain SBOMs will be automatically updated whenever a supplier changes their products.
  - Automation and interoperability for SBOMs will be important for its adoption and wide-scale implementation.
  - SBOMs will become widespread across multiple industries.
- SBOM use cases will extend into OT (operational technology) sector to drive critical infrastructure security.

WITH REZILON'S DYNAMIC SBOM, customers know their real attack surface as it changes dynamically. The Dynamic SBOM allows customers to understand true risk with built-in vulnerability context and VEX. It enables organizations to search for vulnerable components and use that information to take appropriate remediation action throughout the development lifecycle - all in real time. Additionally, customers can control their supply chain risk, achieve compliance and share the Dynamic SBOM by exporting it as a CycloneDX artifact.

Rezilion's Dynamic SBOM seamlessly plugs to all software environments, from development to production, and provides real-time visibility to all software components. It 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. This provides organizations with an unparalleled solution to understand where bugs exist – but also whether or not they could be exploited by attackers.

CASE STUDY: **Rezilion's dynamic SBOM can help customers manage their Log4j challenges in three easy steps.**

STEP 1: Create a dynamic SBOM and know all software components present, search, and discover Log4j instances in your environment.

STEP 2: Establish the exploitability context and know which of the present instances are actually exploitable.

STEP 3: Remediate by applying the most up-to-date patches without causing any operational risks.

→ The complete case study "Using a Dynamic SBOM to Discover Log4Shell is available to download at: **https://www.rezilion.com/using-dynamic-sbom-to-discover-log4shell/**

Rezilion's Dynamic SBOM is available now, free-of-charge for use, in CI environments such as Gitlab. For more information, visit https://www.rezilion.com/platform/dynamic-sbom/ and to sign up for a free 30-day trial at https://www.rezilion.com/get-started/.

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.