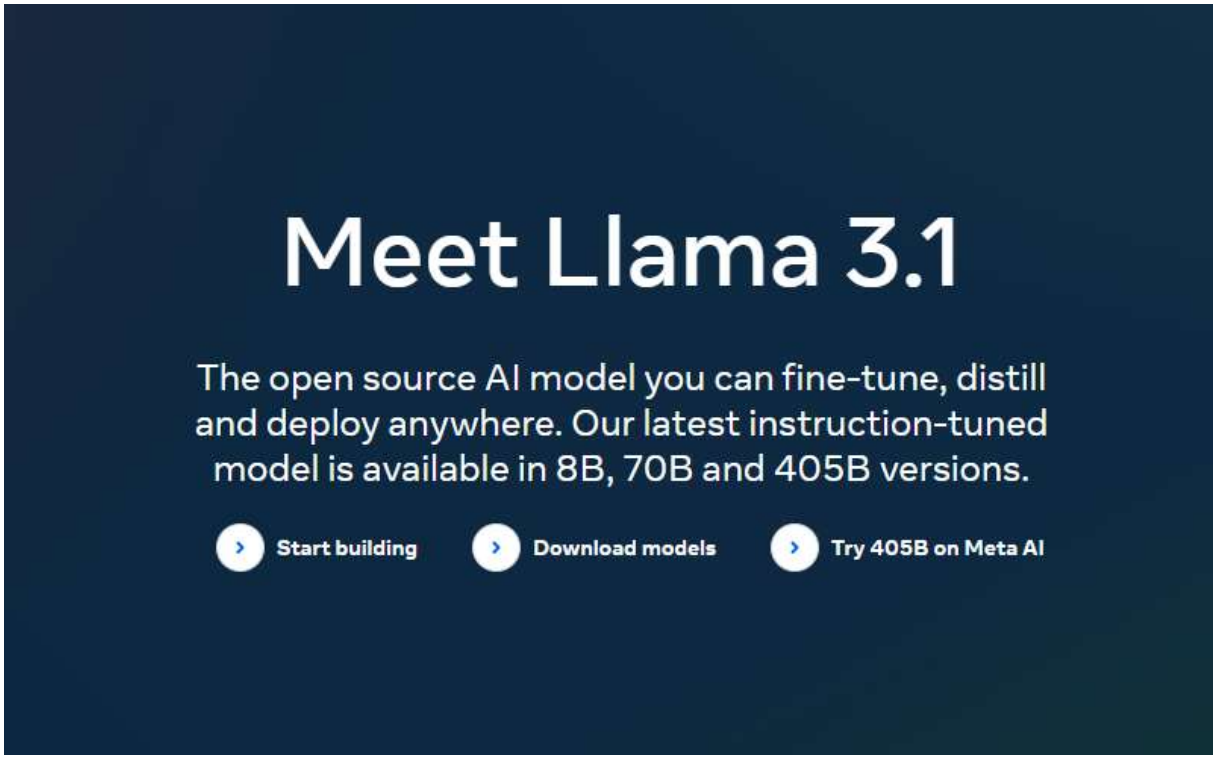
Headline

Open-source SW management plan using software bills of materials (SBOM)

Hae-beom Lee, Senior Consultant / Financial Consulting Team 1

■ Overview

In the 1990s, companies built IT systems and started related businesses. At that time, software (SW) development was mostly the domain of professionals with specialized knowledge, with only a small number of enthusiasts who studied on their own. But by the mid-2000s, the IT industry had expanded, and the world was becoming more connected through the Internet and mobile technology, making it possible for anyone with a little knowledge or interest to develop software programs. As of 2024, we have entered a world where AI automatically generates software, something that once only seemed possible in SF movies. The key that makes all this possible is 'open-source SW.'

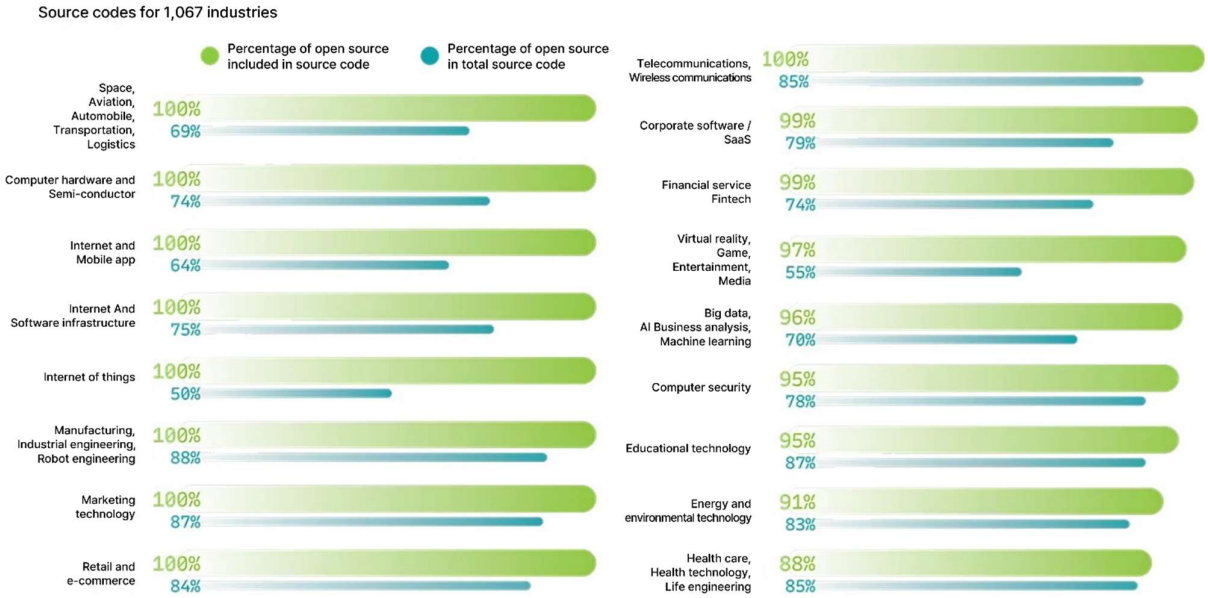


* Source: Meta Llama

Figure 1. Llama 3.1 – Open-source AI language model

■ Software supply chain and open-source SW

There was a time in the past when the cost of purchasing software was very burdensome. But there were also pioneers who believed that IT should not be monopolized by a small number of companies or individuals. They used their talents to develop software and made it available to the public for free. The open source community that has since developed on the Internet has provided easy education, rapid development, and significant cost savings. As a result, open-source SW is widely used in most businesses today.



* Source: 2024 Open Source Security and Risk Analysis Report (https://www.synopsys.com)

Figure 2. Open source usage trends

Traditionally, software was created and supplied from start to finish by a single or a small number of organizations or companies. As open-source SW developed, however, the concept of software supply chain management emerged.

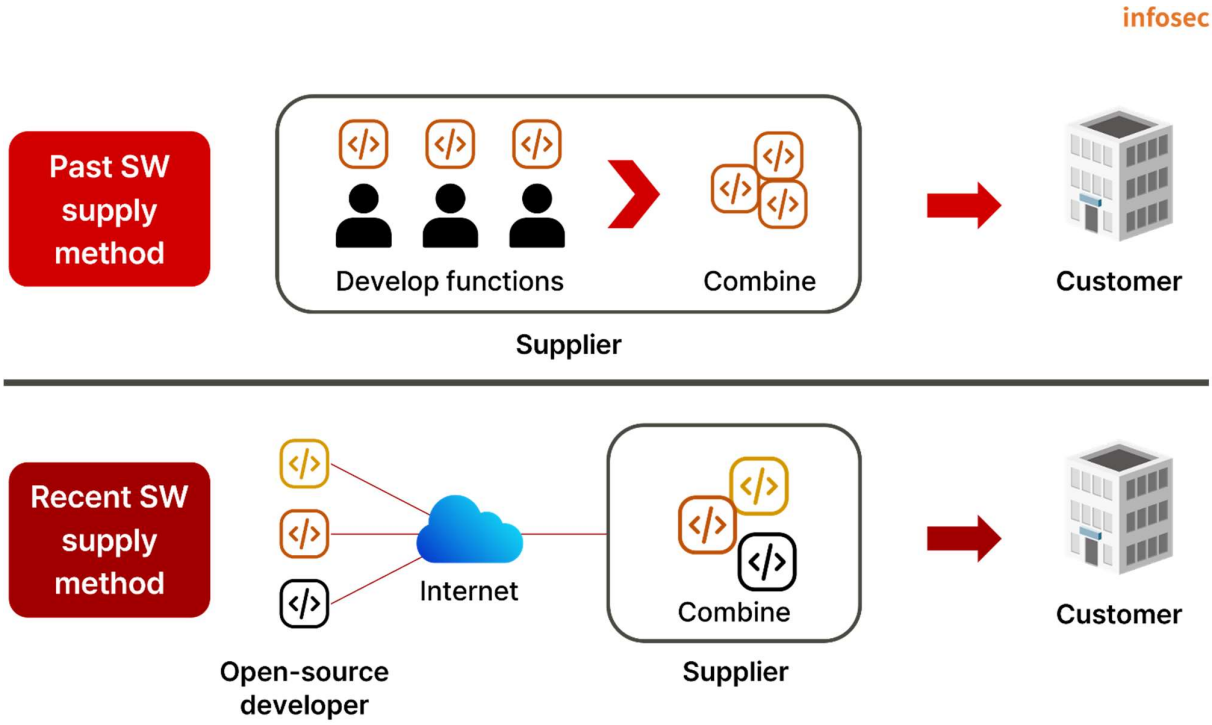
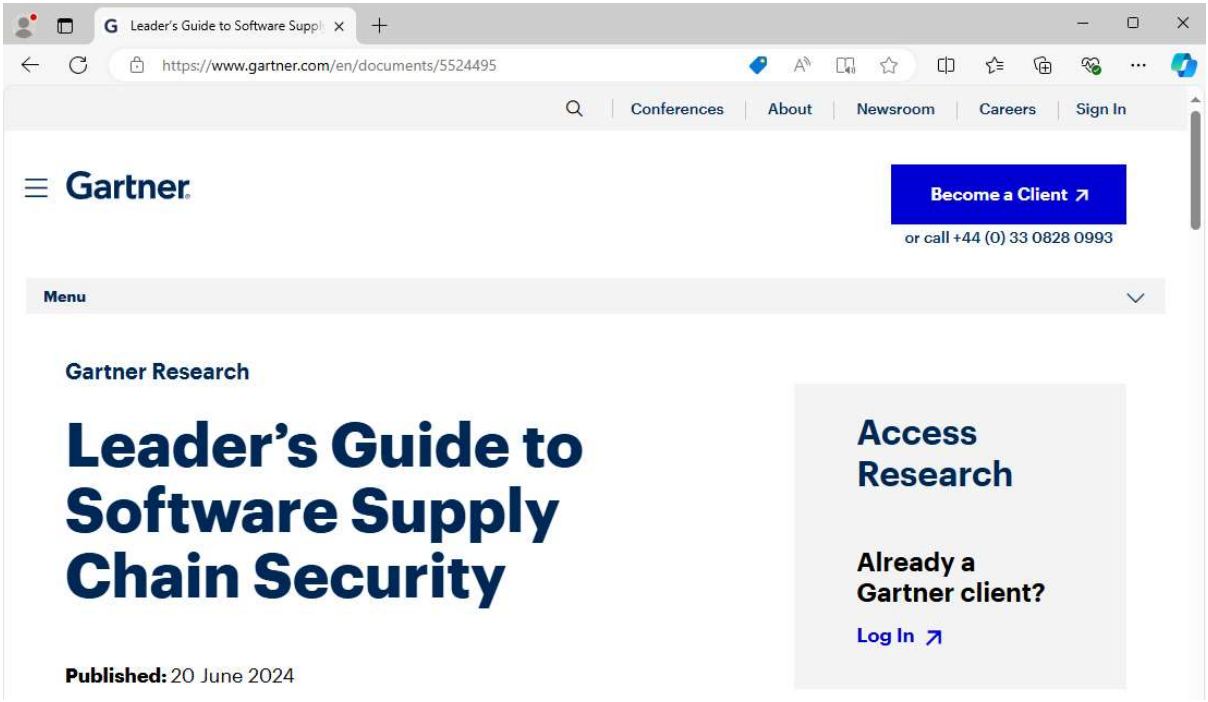


Figure 3. Concept of the software supply chain

As more and more entities and software participate in the software supply chain, the difficulty of management also increases. Recently, the number of security breaches exploiting vulnerabilities of open-source SW has been increasing continuously. The 'Leader's Guide to Software Supply Chain Security' published in June 2024 by Gartner predicts that the following problems will arise as the use of open-source SW increases:

1. The damage from attacks on software supply chains is expected to increase from $46 billion in 2023 to $138 billion in 2031.

2. More than 90% of software used by businesses and institutions contain open source dependencies, and 74% contain high-risk dependencies.



* Source: Gartner website

Figure 4. Leader's Guide to Software Supply Chain Security

■ Efforts to secure the software supply chain

As open-source SW is already widely used, attacks on the software supply chain are no longer the problem of individual persons or companies. These attacks have implications for the entire nation and society, including governments, businesses, organizations and industries that use relevant open-source SW.

Attack (year)	Accident and damage details
SolarWinds (2020)	A Russian-based hacking group compromised the software development environment and distribution systems of IT software vendors, affecting more than 18,000 organizations.
CodeCov (2021)	An attacker exploited a container image security vulnerability to leak credentials for the deployment environment for source code verification, affecting approximately 29,000 customers worldwide.
Colonial Pipeline (2021)	A ransomware that infected the IT systems of a pipeline operator caused a supply disruption across 8,900 km of the southeastern United States, leading to a declaration of emergency in the area and the payment of approximately KRW 5 billion in ransom money.
Log4Shell (2021)	An attacker exploited a zero-day vulnerability in Log4j and publicly available proof-of-concept code to plant malware, and carried out a massive hacking attack targeting vulnerable servers around the world.
Kaseya (2022)	An attacker hacked cloud-based IT remote management solution servers and distributed ransomware disguised as update files to customers, causing damage to approximately 1,500 organizations in 17 countries.
3CX (2023)	An attacker infected PCs that downloaded X-Trader financial software with malware, spreading it to more than 600,000 customers and 12 million organizations.
INISAFE (2023)	Attackers exploited security vulnerabilities in financial security authentication software to hack PCs and spread malware, causing damage to 61 domestic organizations and a total of 207 institutional, corporate and personal PCs.

* Source: Ministry of Science and ICT, SW Supply Chain Security Guidelines (May 2024)

Table 1. Major breaches in the software supply chain

Recognizing the seriousness of security, countries around the world are implementing systems to establish and comply with regulations and guidelines for software supply chain security.

Country	Policies and systems
USA	[May 2021] Executive order (EO 14028, May 2021) of Biden administration • To provide SBOMs for software supplied to the federal government [Mar. 2023] Cybersecurity for medical devices strengthened by Food and Drug Administration (FDA) • All manufacturers requesting approval from the FDA shall provide an SBOM that includes a list of the device's open-source and commercial software components.
EU	[Sept. 2022] Cyber Resilience Act (CRA) bill proposed • SBOMs must be submitted for digital devices supplied (distributed) in the EU.
Japan	[May 2022] SW TF established within the Ministry of Economy, Trade and Industry. • The SW TF established in the Ministry of Economy, Trade and Industry conducts SBOM verification (PoC) for projects in the medical, automobile and software fields.

* Source: Ministry of Science and ICT, SW Supply Chain Security Guidelines (May 2024)

Table 2. Country-specific policies and systems for protecting the software supply chain

The way people think about a particular problem is similar, regardless of nationality. If we look at the references in the above material, **one common solution is the software bill of materials (SBOM)**.

■ Emergence of SBOMs

Research on SBOM standards is active abroad, and such standards have also been established in Korea. Because most of the items are similar across standards and it is difficult to determine which standard is best, companies can choose and use the standard that best suits their situation.

Standard format	Description
SPDX® (Software Package Data Exchange	This was developed by the Linux Foundation in 2011 and registered as an international standard in 2021 (ISO/IEC 5962:2021). • It facilitates open source license management and the use of the SBOM format, and conveys component, license, copyright and security information related to software packages.
CycloneDX (CDX)	Developed by the OWASP community, this standard aims to be a full-stack BOM industry standard that supports supply chain functions. The initial prototype was released in 2017, and the latest standard is version 1.4. • Designed specifically for the SBOM format from the beginning, it supports a variety of specifications, including SaaS BOM.
SWID (Software Identification)	This standard was published by NIST in 2009 and registered as an international standard (ISO/IEC19770-2:2015) in 2015. • It contains information about specific releases of software products and supports inventorying for commercial and open-source SW installed on devices by creating tags for software information.
TTAK.KO-11.0309	This is the software bill of materials (SBOM) attribute standard for open software supply chain management established by the Korea Telecommunication Technology Association (TTA). • It presents 15 SBOM management items for variable SBOM management according to various software supply chains and usage purposes.

* Source: Electronics and Telecommunications Research Institute, SW Supply Chain Management and SBOM Trends (Aug. 2023)

Table 3. Domestic and foreign SBOM standards

The way the SBOM is displayed and its name are slightly different between standards. The National Telecommunications Information Administration (NTIA) of the United States has suggested the minimum items required for an SBOM and the relationships between standards.

Attribute	SPDX	CycloneDX	SWID	TTAK.KO-11.0309
Author Name	(2.8) Creator:	metadata/authors/ author	<Entity> @role(tagCreator), @name	ComponentAuthor:
Timestamp	(2.9) Created:	metadata/timestamp	<Meta>	ReleaseDate:
Supplier Name	(3.5) PackageSupplier:	Supplier Publisher	<Entity> @role (softwareCreator/ publisher), @name	ComponentSupplier:
Component Name	(3.1) PackageName:	name	<softwareIdentity> @name	ComponentName:
Version String	(3.3) PackageVersion:	version	<softwareIdentity> @version	ComponentVersion:

Component Hash	(3.10) PackageChecksum: (3.9) PackageVerification Code:	Hash "alg"	<Payload>/../<File> @[hash-algorithm]:hash	FileChecksum:
Unique Identifier	(2.5) SPDX Document Namespace (3.2) SPDXID:	bom/serialNumber component/bom-ref	<softwareIdentity> @tagID	FormatID:
Relationship	(7.1) Relationship: DESCRIBES CONTAINS	(Inherent in nested assembly/subassembly and/or dependency graphs)	<Link> @rel, @href	IncludeComponent, ImportComponent

* Source: Framing Software Component Transparency: Establishing a Common Software Bill of Materials (SBOM) (Oct. 2021)

* Source: Electronics and Telecommunications Research Institute, SW Supply Chain Management and SBOM Trends (Aug. 2023)

Table 4. Data attributes between SBOM standards

■ Necessity of SBOMs for open-source SW management

Some materials explain SBOMs by referencing the ingredient labels attached to food, but in fact, SBOMs are a more important concept than that. A better analogy would be a recipe for a popular restaurant's secret sauce. This recipe is so important that if it were leaked to a competing restaurant, it could have a serious negative impact on the restaurant's business.

One day, customers who eat the secret sauce begin to get stomachaches, causing the restaurant's reputation and sales to decline. While checking the list of ingredients used in the secret sauce and where they were purchased, the restaurant owner discovers that there is a problem with the ingredients supplied from a factory that suffered a power outage a few days before. To resolve the issue, the owner immediately re-sources the same ingredients from another supplier, thus regaining the trust of his customers.

In the past, software was generally developed from start to finish by a single or small number of developers or companies, but today, the use of open-source SW has become common. When a security incident occurs due to open-source SW, quickly finding the cause of the problem and establishing security measures is no longer an additional task, but an essential task. For software supply chain management, SBOM management is essential in order to address security vulnerabilities and secure customer trust.

It may be unfair from the perspective of open-source SW, but it is considered a target of management and surveillance worldwide for the following three reasons:

1. Security vulnerability issues

When a security vulnerability is discovered, the manufacturer or supplier of commercial software kindly provides security patches or countermeasures, but in the case of open-source SW, users must find and resolve the problem themselves.

2. License issues

Open-source SW is provided free of charge, but to use it for free, users must check the license conditions themselves and take action accordingly.

3. Reputation issues

Sometimes, security may be questioned simply because the software was developed by a specific group.

Managing the SBOMs of software or applications used in an organization or enterprise can help resolve some of the issues that may arise when using open-source SW.

1. Rapid identification of and response to security vulnerabilities

In the case of commercial software, when a security vulnerability is discovered, the supplier analyzes it on its own and provides a solution to the customer. On the other hand, in the case of open-source SW, users must check for security vulnerabilities and resolve them themselves. Systematically managing SBOMs allows the user to quickly identify open-source SW in use and quickly establish countermeasures against security vulnerabilities.

2. License identification and appropriate actions

Commercial software rarely has licensing issues because users pay for it at the purchase and contract stage and obtain licenses that fit their organization's IT operating environment. However, in the case of open-source SW, even if it is provided free of charge, complex license conditions must be complied with. If staff misinterpret the license or fail to identify a problem, it could lead to a risk of legal action. By managing the license information of open-source SW through SBOMs, legal issues can be resolved in advance in the early stages of software development.

3. Review of the security capabilities of the developer/supplier and decision on whether to continue use

In April 2024, the United States legally banned the use of the popular video-sharing platform TikTok. This was out of concern that the developer was a Chinese company and might be asked to provide personal information to the Chinese government. Including information about developers and suppliers in the SBOM will make it easier to implement appropriate security measures when such issues arise, and to replace or discontinue software as needed.

SBOM management plays a critical role in increasing software transparency and effectively managing security, legal issues and supply chain risks.

■ Measures to establish an open-source SW management system

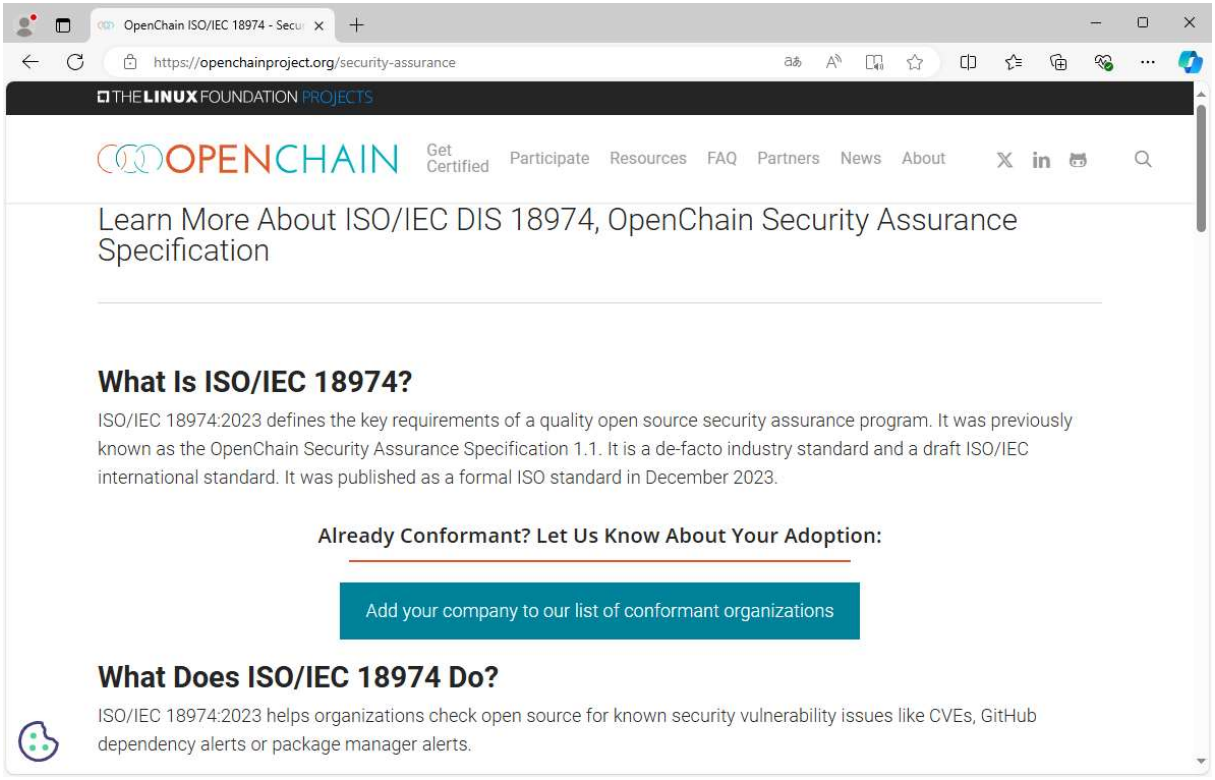
Recently, activities are being carried out in Korea to maintain the security of the software supply chain and manage open-source SW. The financial authorities have published a guide to using open-source SW and regularly hold forums on software supply chain security for financial company officials. In addition, a trend of introducing or reviewing automated systems for SBOM management is spreading, especially among banking institutions.

However, introducing an automated system does not automatically solve all problems. Every system has its own unique characteristics, which can lead to false detection or missed detection issues, and due to the nature of open-source SW (in terms of information protection and legal compliance), the management entity and person in charge of the work must be clearly designated. A risk assessment methodology is also necessary. In order to comprehensively manage these issues, it is essential to establish an open-source SW management system.

In the following sub-sections, the author introduces the main contents of ISO/IEC 18974:2023, an international standard for open-source software management systems, and shares considerations for actually introducing an open-source software management system.

1. ISO/IEC 18974:2023

ISO/IEC 18974:2023 defines the key requirements of a quality open source security assurance program. It is a de-facto industry standard and a draft ISO/IEC international standard. It was published as a formal ISO standard in December 2023.



* Source: ISO/IEC 18974:2023

Figure 5. ISO/IEC 18974:2023

ISO/IEC 18974:2023 presents management plans for the following three items:

1. The key places to have security processes
2. How to assign roles and responsibilities
3. And how to ensure sustainability of the processes

Companies wishing to implement an ISO/IEC 18974:2023 management system can go to this site to download an open-source self-certification checklist of requirements for ISO standards. For those who need to establish an open-source SW management system but haven't decided how to do it, the following checklist is recommended:

Section	Requirements
4.1.3	• Program participants are aware of the open source security assurance policies and where they can find them. • Program participants are aware of goals of the relevant open-source software. • Program participants are aware of the expected contributions to ensure the effectiveness of the program. • Program participants are aware of the consequences of not following program requirements.

4.1.4	• We have a written statement that clearly defines the scope and limits of the program. • There is a set of metrics to measure program performance. • We have documented evidence from each review, update or audit to demonstrate continuous improvement.
4.1.5	• We have a way to identify structural and technical threats to the software provided. • We have a way to detect the presence of known vulnerabilities in the software provided. • We have a way to follow up on the identified know vulnerabilities. • We have a way to communicate the identified known vulnerabilities to our customer base when assurance is required. • We have a way to analyze the supplied software for known vulnerabilities newly published after the release of the software. • We have a way to apply continuous and repetitive security testing to all software supplied prior to release. • We have a way to ensure that identified risks have been removed before the software supplied is released. • We have a way to appropriately deliver information about identified risks to third parties.
4.2.1	• We have a mechanism that allows third parties inquire about known or newly discovered vulnerabilities (e.g., via an email address or web portal monitored by program participants). • We have documented internal procedures for responding to third-party inquiries about known or newly discovered vulnerabilities.
4.2.2	• We have documented the people, groups, or functions related to the program. • We have ensured that the identified program roles have been appropriately staffed and funded. • We have ensured the available expertise to address the identified known vulnerabilities. • We have documented procedures for assigning internal responsibility for security assurance.
4.3.1	• We have documented procedures to ensure that all open-source software used in the provided software is continuously recorded throughout the life cycle of the provided software. This includes an archive of all open-source software used in the provided software. • We maintain records of open-source components for the software provided which demonstrate that documented procedures have been properly followed.
4.3.2	• We have documented procedures for handling the detection and resolution of known vulnerabilities in open-source SW components of the software provided. • We maintain records of open-source components for the software we provide, which allows us to track the known vulnerabilities identified and actions taken (even when no action was required).
4.4.1	• We have documented evidence that the program meets all requirements of this specification.
4.4.2	• We have documented evidence that we have reviewed the program for adequacy within the past 18 months.

* Source: ISO/IEC 18974 Online Self-Certification Checklist

Table 5. ISO/IEC 18974 Online Self-Certification Checklist

2. Considerations for the introduction of an open-source SW management system

1) Organizational structure

Companies adopting open-source SW have very diverse organizational structures and service goals. Management objectives vary depending on whether it is for external services or internal business systems, and management should be undertaken by the organization that can best manage the objectives to achieve successful results. However, it is difficult to guarantee the successful operation of the management system through the organizational structure alone. In order to operate a successful management system, each organization must play an active role. In addition, cooperation between organizations must be achieved through a consultative body or communication channels.

When used for internal business purposes, open-source SW is usually operated in a closed environment, so the possibility of security vulnerabilities or licensing issues occurring is

relatively low. Therefore, in such an environment, where rapid development and rapidly reflecting requirements are important, it may be more effective for the development organization that directly handles open-source SW to be the management entity. On the other hand, if open-source SW is used for external services, there is a high risk of direct attacks due to exposure to security vulnerabilities, and licensing issues may arise due to the exposure of the program. In such cases, it is appropriate for the IT planning organization to operate the management system.

Service type	Management goal	Organization	Role
For internal business	To rapidly reflect business requirements	Development	Operation of the management system, SBOM management
		Information security	CVE vulnerability management
		Legal	License management
		IT planning	Review
For external services	To stably provide services	IT planning	Operation of the management system, SBOM management
		Information security	CVE vulnerability management
		Legal	License management
		Development	Reputation management

Table 6. Management goals and organizational roles by service type

2) Introduction of the system

Small service organizations may have difficulty deciding where to start with open-source software management. However, many sites in Korea provide a variety of information related to open-source SW, and useful results can be found when searching for information on security vulnerabilities or licensing issues.

Site name	URL	Service
Open SW portal	www.oss.kr	• Retrieving open-source SW security vulnerability information
Open-source SW License Comprehensive Information System	www.olis.or.kr	• Retrieving open-source SW license information

Table 7. Sites providing open-source SW information

If a large number of services use open-source SW or involve various stakeholders, it may be necessary to introduce an automated management system. There are several automated systems for open-source SW management, and there are many more systems in addition to the ones introduced below, so choose the one that best suits the organization and service.

System name	Characteristics
Black Duck	• A comprehensive solution for managing licenses, vulnerabilities and source code quality while using open-source SW • Licensing and security management for open-source SW throughout the software supply chain and application life cycle
LABRADOR	• Software supply chain security management supported through SBOMs containing open-source SW components • A software safety management platform that can detect and patch license and vulnerability risks in open-source SW
FOSSID	• A license and security vulnerability management solution for open-source SW. It detects components in the source code and identifies the licenses and security vulnerabilities of each component. • Provides a massive open source database, automatic data collection technology and AI-based improved detection performance.
White Source	• Provides license compliance and vulnerability management services based on a massive database, and supports various environments such as container and serverless.
Sparrow SCA	• A tool that identifies open-source SW licenses and diagnoses security vulnerabilities. • Provides support for source code and binary file analysis, and snippet analysis that imports only a part of the open-source SW source code.

* Source: Financial Supervisory Service, Guide to Utilization and Management of Open-source Software in the Financial Sector (Dec. 2022)

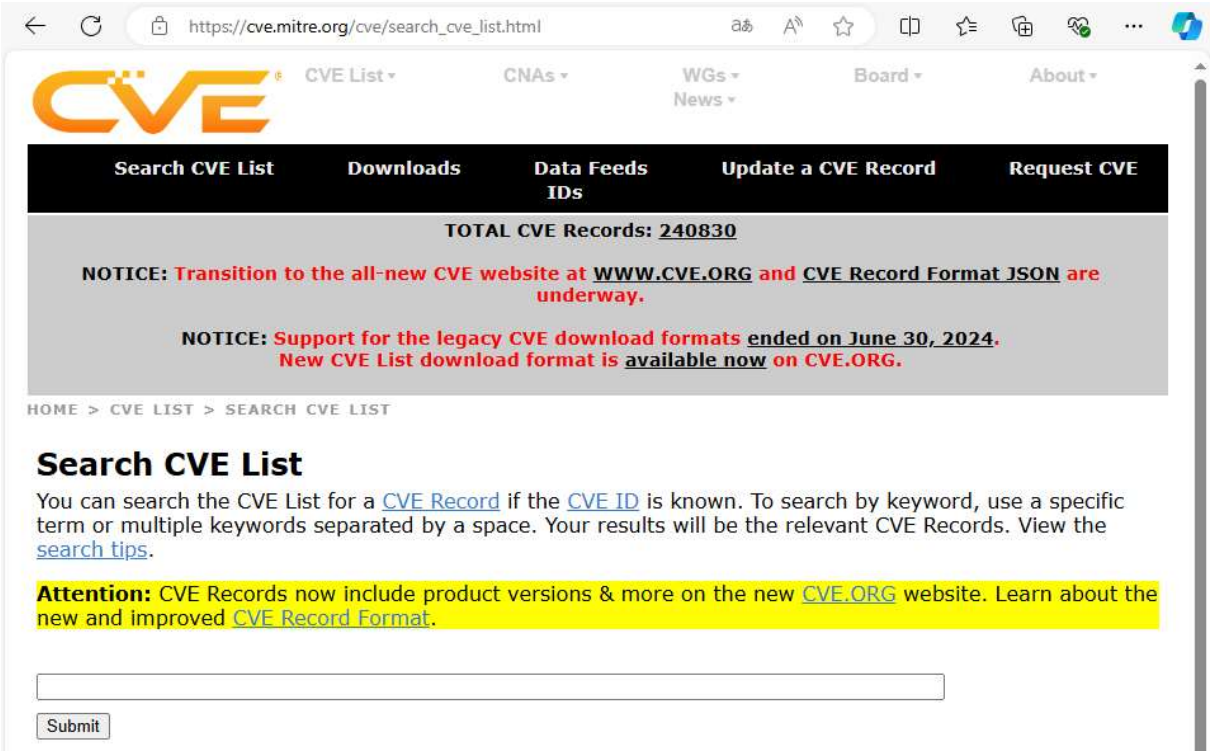
Table 8. Open-source SW automation management system

3) SBOM management items

A. Selection and utilization of items for security vulnerability management (CVE search method)

The most important task is to identify technical vulnerabilities in open-source SW. However, if no accident occurs due to a vulnerability, it is difficult to take action, and it is often overlooked or not recognized.

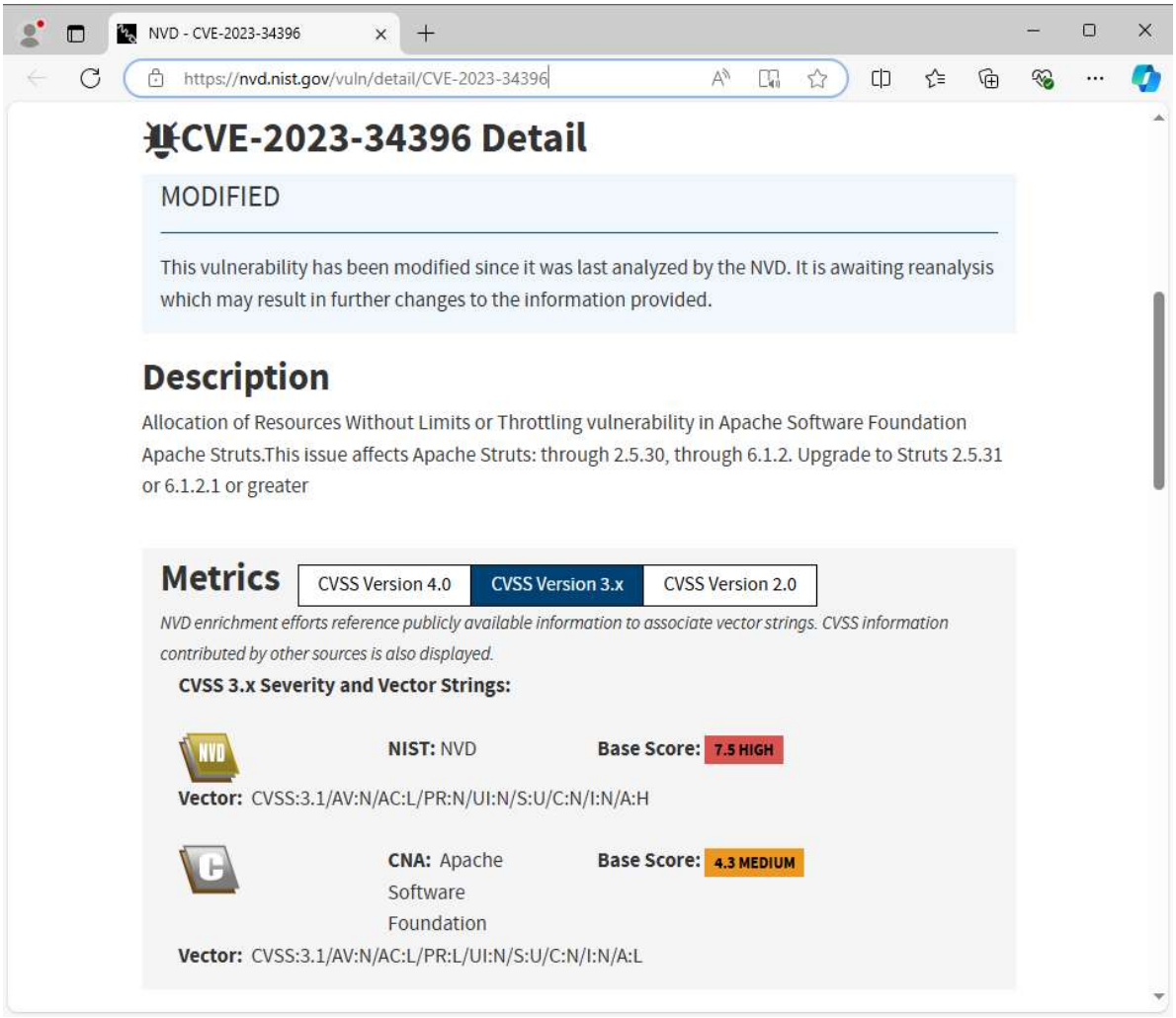
The easiest way to check for security vulnerabilities is to search in the Common Vulnerabilities and Exposures (CVE) page of the American non-profit organization MITRE.



* Source: MITRE

Figure 6. CVE search page

Clicking on a CVE ID in the search results will provide an overview of the vulnerability and reference links. For reference, the National Vulnerability Database (NVD) link shows the Common Vulnerability Scoring System (CVSS) score for that vulnerability, which can be used to assess the severity of the security vulnerability.



* Source: NIST

Figure 7. CVSS search page

Key Point – SBOM items for security vulnerability management: Name and version

B. Selection and utilization of items for license management (measures by license)

Although licenses are not technical security vulnerabilities, they are important items that can result in financial losses at the level of a security incident if not managed properly. In general, license information is posted on the website of the organization that developed the open-source SW. Because open-source SW licenses vary greatly in type and usage conditions, it is necessary to identify the correct license. In addition, some licenses require disclosure of all source code associated with the open-source SW, so be careful when deciding whether to use them.

Category	License	Key terms of use
Permissive (Allowed)	Apache-2.0, BSD-2-Clause, BSD-3-Clause MIT	• Indicate copyright and license

Weak Copyleft (Weak constraint)	LGPL-2.1, LGPL-3.0, EPL-2.0, MPL-2.0	• Partially disclose the source code that uses the open-source code
Copyleft (Strong constraint)	GPL-2.0, GPL-3.0	• Disclose all the source code combined with the open-source code

Source: Ministry of Science and ICT, SW Supply Chain Security Guidelines (May 2024)

Table 9. License categories and key terms of use

Key Point - SBOM items for license management: Type and name of license

C. Selection and utilization of items for reputation management

Apart from the technical point of view, assessing whether the open-source SW is trustworthy and safe to use continuously is a subjective matter.

When making purchases in our daily lives, we usually choose products based on criteria such as trustworthy manufacturer, widely used product, robust product and continuous after-sales service. Similar criteria can be applied when selecting open-source SW. In other words, it is advisable to select software based on criteria such as software managed by a well-known IT company or foundation, software supported by many developers and an active community, and software managed by a trustworthy country.

Key Point - SBOM items for reputation management: Manufacturer and country

4) Risk assessment

It is not realistic to perfectly address all security vulnerabilities. In order to objectively judge and decide the level of management to apply to open-source SW, it is necessary to establish risk assessment criteria for security vulnerabilities.

To manage the security vulnerabilities, licenses and reputations decided on above, vulnerability assessment criteria can be established as follows:

Assessment item	Example of assessment criteria	Remark
Security vulnerability	CVE exists, and CVSS 9.0 – 10.0	Establish criteria according to the severity classification criteria "Low," "Medium," "High," and "Critical" based on CVSS v3.x calculation results (Source: https://nvd.nist.gov)
	CVE exists, and CVSS 7.0 – 8.9	
	CVE exists, and CVSS 4.0 – 6.9	
	CVE exists, and CVSS 0.1 – 3.9	
License	Copyleft (strong constraint) license to be used	Establish criteria based on the technical difficulty of measures to meet license usage conditions
	Weak copyleft (weak constraint) license to be used	
	Permissive (allowed) license to be used	

Reputation	Management entity/individual, etc., not trusted	Establish criteria based on the size, community accessibility, reliability, etc., of the management entity
	No management entity, but active community	
	Managed by global IT company, IT foundation, or major domestic IT company, etc.	

Table 10. Example of vulnerability assessment criteria

It is possible to perform consistent risk assessments by selecting a score calculation method and risk calculation formula according to the vulnerability assessment criteria based on the risk assessment methodology established by each company.

It is not practical to take action against all risks. To take more effective measures against risks, it is necessary to select an acceptable degree of assurance (DoA) to determine the acceptable level of risk according to the basic risk management methodology, and then take measures for risks that exceed this level.

In general, risk management plans classify risk management into four types: risk reduction, risk transfer, risk avoidance and risk acceptance. For effective risk management, it is important to establish a countermeasure plan for each vulnerability type.

Measures	Security vulnerability	License	Reputation
Risk reduction	• Apply security patches	• Reflect the license use conditons - Indicate copyright, open related source, etc.	• Replace with open-source SW managed by a global IT company or IT foundation
Risk avoidance	• Replace with open-source SW with no related vulnerabilities	• Replace with open-source SW with permissive (allowed) condition • Replace with commercial software	• Replace with open-source SW managed by a global IT company or IT foundation.
Risk transfer	• Sign up for security incident insurance	• Sign up for security incident insurance	• Sign up for security incident insurance
Risk acceptance	• Apply supplementary controls (e.g., firewall)	-	-

Table 11. Measures against open-source SW risks

5) Necessity of managing an open-source SW repository

Open-source SW can be downloaded from anywhere as long as there's Internet access, but it's important to be careful when downloading files from sources other than the official site. This is because using open-source SW containing malicious code in a development project can create vulnerabilities. To use open-source SW safely, it is necessary to build a repository, control unauthorized access and strictly enforce file import procedures. By

additionally managing integrity verification values (HASH) for open-source SW in the SBOM, it is possible to protect against file forgery and modification.

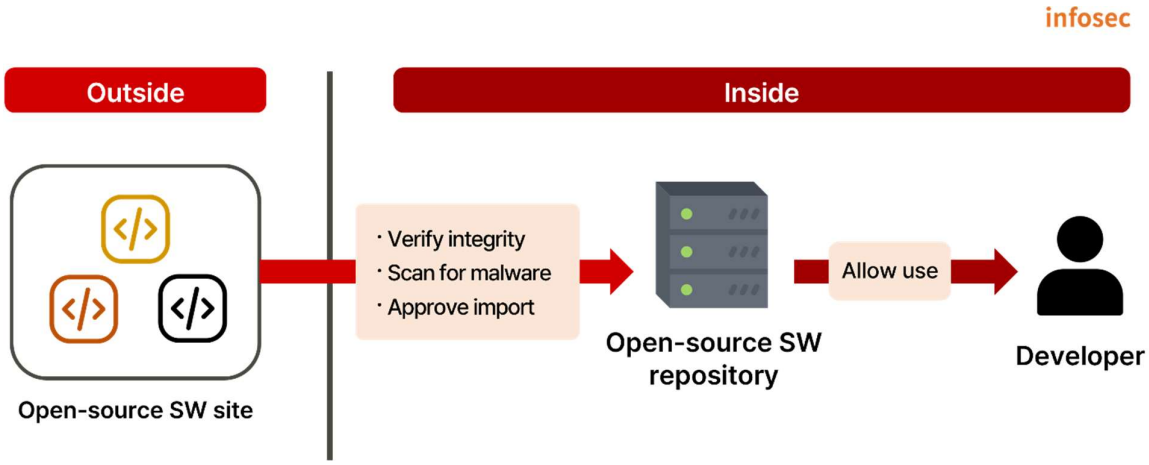


Figure 8. Open-source repository import flow

■ Conclusion

In all industrial sectors of modern society, IT technology has gone beyond simple auxiliary tools to become a key element for value creation. With the advancement of AI, IT technology is being incorporated into areas such as art and emotions that were previously considered to be unique to humans. IT security incidents will have global implications beyond just financial losses to a single organization or service.

Although software bill of materials (SBOM) management cannot completely prevent IT security incidents, it is surely the fastest and most reasonable way to recover after an incident occurs. Even without specialized knowledge of security or IT technology, it's easy to start open-source SW management by checking which open-source SW the company is currently using. The next step is to decide whether to introduce SBOMs.

SK Shieldus provides consulting services in relation to the establishment of open-source SW management systems. For more information, please visit the SK Shieldus website or contact us.