IBIMA Publishing
Communications of International Proceedings
https://ibimapublishing.com/p-articles/46SP/2025/4641625/
Vol. 2025 (37), Article ID 4641625

Integrated Threat Modeling Framework for Software Supply Chain Security*

Piotr KONTOWICZ

Poznan University of Technology, Faculty of Computing and Telecommunications,
Piotrowo 3, 60-965 Poznan, Poland

Correspondence should be addressed to: Piotr KONTOWICZ, piotr.kontowicz@put.poznan.pl

* Presented at the 46th IBIMA International Conference, 26-27 November 2025, Ronda, Spain

Abstract

The increasing frequency of attacks targeting the Software Supply Chain (SSC) has made its security a critical component of organizational cyber resilience. The complexity of modern systems, which rely on components from diverse sources, complicates the identification of the attack surface. High-profile incidents such as SolarWinds, Log4Shell, and XZ Utils illustrate that compromises may occur at any stage of the software lifecycle, including acquisition, distribution, and maintenance.
This paper examines the primary attacker profiles (Advanced Persistent Threats, hacktivists, insiders, and script kiddies) and their respective impacts on supply chain risk. It further discusses established threat modeling methodologies (STRIDE, PASTA, OCTAVE) and supporting risk assessment tools, including CVSS, MITRE ATT&CK, and the Software Bill of Materials (SBOM).
The paper proposes an integrated threat model for the software supply chain that combines attacker profile analysis, attack vector identification, and countermeasure definition within a unified process. This model facilitates repeatable, context-aware risk analysis and supports the development of organizational resilience against the most probable attack scenarios.

Keywords: Software Supply Chain Security, Threat Modeling, Vulnerability , Artificial Intelligence in Cybersecurity; Supply Chain Attack

Introduction

The rising incidence of attacks on the Software Supply Chain (SSC) has made its security a crucial aspect of digital resilience for contemporary organizations. In recent years, supply chain attacks have disrupted operations and resulted in substantial financial losses. Modern applications and systems are composed of numerous interconnected components, frequently incorporating open-source elements, which complicates the definition of the attack surface.
Government agencies and standardization bodies are developing risk management frameworks and practices to enhance awareness of components used within software solutions. Despite existing regulations and recommendations, many organizations encounter challenges in implementing these practices. There is a need for a process that integrates threat analysis with mitigation actions and offers a high degree of automation.

Software Supply Chain: Structure

The software supply chain refers to the entire lifecycle of software creation, distribution, and maintenance, encompassing all elements that contribute to the final product. Software security depends on the trustworthiness of all components within the supply chain.

Cite this Article as: Piotr KONTOWICZ, Vol. 2025 (37) "Integrated Threat Modeling Framework for Software Supply Chain Security " Communications of International Proceedings, Vol. 2025 (37), Article ID 4641625, https://doi.org/10.5171/2025.4641625

Modern software constitutes a complex ecosystem composed of elements from diverse sources, with each component representing a potential attack vector. Consequently, there is an increasing need to formalize the management of supply chain security.

Case Studies of Major Software Supply Chain Incidents

One of the most serious software supply chain attacks was the attack on SolarWinds [1], uncovered in 2020. As a result, approximately 18,000 SolarWinds customers installed a software update containing a hidden backdoor used by attackers to steal data and spy on organizations. The target of the attack was the SolarWinds Orion performance monitoring tool. This software has privileged access to resources, which meant that the attackers who added the backdoor also gained privileged access to SolarWinds customers' systems.
The attack can be divided into several stages:

1. Gaining access to the SolarWinds network
2. Testing code injection into the Orion tool
3. Infecting the Orion software
4. Distributing the compromised version containing the backdoor via official channels

These actions enabled the attackers to distribute an update containing malicious code, which granted access to victim systems and facilitated subsequent activities such as data theft, espionage, and the compromise of additional systems within organizations utilizing the Orion tool.

One widely known vulnerability related to the supply chain is CVE-2021-44228 [2], commonly referred to as Log4Shell. It was discovered in 2021 and is a critical RCE [3] vulnerability found in the Apache Log4j library, which is deeply embedded in the software supply chain and is one of the most widely used open-source libraries in the world. It often appeared as an indirect dependency, meaning companies were not using it directly, but rather through tools provided by other vendors who relied on the library.

The attack mechanism involved sending a specially crafted string by an attacker. This string could be passed in any way that would be logged by the application. When the vulnerable version of the library encountered such a string in the log, it interpreted it as a command, potentially allowing attackers to gain control over the device. The primary mitigation method was updating the Apache Log4j library to a version where the vulnerability had been fixed.

A relatively recent vulnerability, CVE-2024-3094 [4], was linked to a sophisticated long-term attack on the XZ Utils package. It received the highest possible CVSS score because it allowed remote code execution. The attack involved the intentional insertion of malicious code into the package. The entire attack was spread over time, with the infiltration process lasting approximately two years:

1. 2021 – 2023: the attacker began making small contributions to the project, gradually building the trust of the project maintainers. Over time, they offered to help manage the project to relieve the main maintainer,
2. 2023 – 2024: after obtaining co-maintainer status, the attacker eventually became responsible for releasing new versions of the package,
3. February – March 2024: introduction of malicious modifications to the XZ Utils packages.

It is important to note that the malicious modifications were not inserted directly into the source code but were embedded in obfuscated text files within tarball archives. The modification was discovered by Andres Freund, a Microsoft employee, who identified unusual CPU usage and initiated an investigation on March 29. Since the changes had not been incorporated into stable releases, a global-scale infection was averted.

Attacker Profiles and Their Impact on the Software Supply Chain

For effective risk modeling, it is essential to understand the categories of attackers, their motivations, available resources, and operating techniques:

1. Advanced Persistent Threat (APT): an organized group of attackers with significant resources, operating according to a defined strategy, often linked to nation-states or large organizations. Their objectives include espionage, intellectual property theft, and potentially destabilizing operations. These groups are characterized by high technological sophistication and have access to advanced tools, including zero-day

vulnerabilities. Their typical attack chain includes reconnaissance, targeted phishing, exploitation, privilege escalation, lateral movement, and long-term presence in the victim's network.
2. Hacktivists: primarily motivated by ideological reasons, aiming to influence public opinion. They usually lack a structured organization and are made up of many members, most of whom are non-technical. Technical members are responsible for preparing and coordinating campaigns. They typically favor high-visibility techniques such as DDoS attacks, website defacement, and publishing stolen data.
3. Insider: the threat originates from someone with legitimate access to resources — an employee, contractor, or partner — who can, either intentionally or through negligence, cause a security incident. Motivations may include financial gain, revenge, or recruitment by foreign intelligence services. This group has knowledge of internal processes, tools, and the organization's technical capabilities; individuals with administrative privileges are particularly dangerous. Detecting insider activity is more difficult than identifying external attacks, as many actions may resemble legitimate operations.
4. Script kiddie: attackers with limited technical skills who rely on ready-made tools, scripts, and publicly available guides. Their motivation is often curiosity or amusement. While they lack deep expertise, their actions can, under favorable conditions, expose serious security flaws — although in most cases, the impact of their attacks is minimal. In the context of the supply chain, they may reveal vulnerable component versions, publicly disclosed data, or configuration errors.

Distinct attacker profiles introduce varied threats to the software supply chain. Advanced Persistent Threats (APTs) present the highest risk, insiders possess the capability to introduce malicious changes directly, hacktivists may expose weaknesses in solutions, and script kiddies contribute by probing for known, unpatched vulnerabilities.

Frameworks and Tools for Supply Chain Threat Modeling

Threat modeling is the process of identifying and assessing potential threats and system weaknesses, as well as developing appropriate countermeasures. The classical approaches include:

1. STRIDE: a methodology developed by Microsoft that focuses on identifying threats such as spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege. This approach helps structure the process of identifying potential attack vectors.
2. PASTA: a risk-focused methodology. Unlike technical models, it integrates the business perspective and organizational context as the starting point. It emphasizes the attacker's perspective by simulating likely attack vectors against identified assets.
3. OCTAVE: a methodology for information security risk assessment. Its core principle is that risk evaluation should be conducted from the company's point of view, with a focus on identifying the organization's most critical assets.

A key component of risk management is assessing the threats associated with identified vulnerabilities. The Common Vulnerability Scoring System (CVSS) is an open framework for quantitatively evaluating security flaws. The score ranges from 0 to 10 and, in its latest version, is based on four metric groups:

1. Base metrics: represent inherent, unchanging characteristics of a vulnerability.
2. Temporal metrics: reflect factors that change over time, such as the availability of an exploit.
3. Environmental metrics: allow the score to be tailored to the context of a specific organization.
4. Supplemental metrics: provide additional context not used in the numerical score.

CVSS is closely linked to the Common Vulnerabilities and Exposures (CVE) [5] system, which assigns unique identifiers to vulnerabilities in specific products, and to the Common Weakness Enumeration (CWE) [6], which categorizes the types of software weaknesses.

Another crucial element in threat modeling is the MITRE ATT&CK framework [7], a global standard for describing adversary tactics, techniques, and procedures (TTPs). It enables the tracking of how attacker behavior evolves — a key aspect of conducting effective threat modeling.

The Software Bill of Materials (SBOM) is essentially a list of software components and is considered a foundational element in improving software supply chain security. Government agencies and regulatory bodies clearly emphasize that it is a key component for risk management, vulnerability assessment, and incident response. When a vulnerability is disclosed in a specific version of a widely used library, having an SBOM allows an organization to quickly determine whether it is potentially affected by the threat. It is important to stress, however, that the mere presence of a vulnerable library in a product does not automatically mean the product is insecure —

additional conditions often need to be met, such as calling a specific function or having a particular environment configuration.

In short, knowledge of using a vulnerable library highlights the need for risk evaluation, verification of the vulnerability's impact on the product, and implementation of appropriate mitigation measures.

Contextual Threat Model for SSC Security

The data from the previously described sources can be integrated into a unified process. The outcome of this process is a more tailored threat model for an organization's software supply chain. The process consists of three steps:

1. Attacker Profile Analysis: Identifying who the potential attacker is, what resources they possess, and what motivates their actions. Depending on the organization conducting the risk assessment, the attacker profiles considered may vary, as will the types of resources that should be taken into account. An example of such an analysis is provided in Table 1.

2. Attack Vector Analysis: Once potential attackers and their techniques have been identified, this information can be mapped onto the supply chain. This contextualization gives meaning to each threat and enables the assessment of potential consequences. In the next step, this also supports the development of risk mitigation recommendations. The software supply chain (SSC) can be divided into four stages:

    a. Acquisition: obtaining components, libraries, hardware
    b. Production: integration and compilation of software
    c. Distribution: delivering software or new versions to the customer
    d. Operation: maintaining software in the customer's environment
3. Definition of Countermeasures: The final step defines specific mitigation actions for each identified and categorized threat.

Table 1. Simplified threat actor analysis across organization

Type of Attacker	Motivation	TTPs for Software	TTPs for Hardware	TTPs for Data
APT	Espionage, Sabotage	CI/CD pipeline attacks to insert trojans	Firmware modification introducing vulnerabilities	Attacks on data processing and storage systems
Cybercriminal	Financial gain	Injection of malicious code into open-source libraries	Exploiting known vulnerabilities in hardware and firmware	Data exfiltration and resale
Insider	Sabotage, financial gain, revenge against employer	Deliberate introduction of vulnerabilities, actions on behalf of APTs	Tampering with trusted components during manufacturing	Data exfiltration, sale to competitors
Industrial competitor	Technology theft	Reverse engineering of code	Reverse engineering of devices	Commissioning data theft by employees or hiring spies within the organization

Towards an Automated and Repeatable Threat Modeling Framework

In comparison to existing approaches, the proposed model provides a repeatable process, which is currently absent in many organizations. Instead of treating individual elements as isolated formal requirements, these components can be integrated into a unified threat modeling process. This integration enables organizations to develop targeted resilience against the most probable adversaries.

Threat analysis largely depends on Cyber Threat Intelligence (CTI), which often comes in an unstructured format. Natural Language Processing (NLP) tools and large language models show strong potential in automating the first stage of the process. The subsequent steps can be automated through data correlation. The previously discussed elements can be cross-referenced with threat feeds such as public vulnerability databases, commercial sources, and exploit probability data. More advanced solutions may use machine learning to simulate multi-stage attacks based on up-to-date information about a given product or organization.

Future research will prioritize empirical validation of the proposed model and investigate opportunities for automating the entire process, enabling risk analysis to be conducted automatically using diverse data sources.

Conclusions

The examined cases confirm that software supply chain security is among the most critical components of organizational resilience against cyberattacks. Incidents such as SolarWinds, Log4Shell, and XZ Utils demonstrate that attacks may occur at any stage of the software lifecycle. Modern applications, constructed from numerous dependencies, create a complex ecosystem with a challenging-to-define attack surface. The diversity of threats posed by various attacker groups necessitates a multifaceted approach.

The proposed model facilitates the integration of all essential elements into a unified process, enabling organizations to develop targeted resilience against attacks.

Future research will aim to automate the risk analysis process by integrating data from both public and commercial sources. Employing natural language processing and artificial intelligence techniques may enable the automatic processing of current threat information and facilitate dynamic, real-time risk assessment. This approach will support the development of adaptive threat models that respond to evolving threats within the software supply chain.

Acknowledgements

This research was funded by the Polish Ministry of Science and Higher Education under Grant 0313/SBAD/1311.

Bibliography

- “SolarWinds Supply Chain Attack,” Fortinet. Accessed: Nov. 04, 2025. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/solarwinds-cyber-attack
- “CVE Record: CVE-2021-44228.” Accessed: Nov. 04, 2025. [Online]. Available: https://www.cve.org/CVERecord?id=CVE-2021-44228
- “Remote Code Execution (RCE) Explained in Detail | Splunk.” Accessed: Nov. 04, 2025. [Online]. Available: https://www.splunk.com/en_us/blog/learn/rce-remote-code-execution.html
- “CVE Record: CVE-2024-3094.” Accessed: Nov. 04, 2025. [Online]. Available: https://www.cve.org/CVERecord?id=CVE-2024-3094
- “CVE: Common Vulnerabilities and Exposures.” Accessed: Nov. 05, 2025. [Online]. Available: https://www.cve.org/
- “CWE - Common Weakness Enumeration.” Accessed: Nov. 05, 2025. [Online]. Available: https://cwe.mitre.org/
- “MITRE ATT&CK Applications in Cybersecurity and The Way Forward.” Accessed: Nov. 05, 2025. [Online]. Available: https://arxiv.org/html/2502.10825v1