Advanced Persistent Threats based on Supply Chain Vulnerabilities: Challenges, Solutions & Future Directions

Zhuoran Tan, *Student Member, IEEE,*, Shameem Puthiya Parambath, *Member, IEEE,*, Christos Anagnostopoulos, *Member, IEEE,*, Jeremy Singer, and Angelos K. Marnerides, *Senior Member, IEEE*,

***Abstract*—Due to the ever increasing inter-dependency across a variety of diverse software and hardware components in Information and Communications Technology (ICT) provisioning, Supply Chain Vulnerabilities (SCVs) targeting such dependencies have evolved as a primary choice for malicious actors to stealthy and complex cyber-attacks. The current *modus operandi* in the cyber threat spectrum is solely correlated with Advanced Persistent Threats (APTs) that have shown to be prevalent across diversified attacks underpinning cyberwarfare, and cybercrime. Hence, defense against such threats is undoubtedly considered as a high priority on a global scale. Nonetheless, the reliance on third-party supply chain software and device across diverse ICT ecosystems, combined with the current defense mechanisms' inability to identify specific compromised entry points, results in an increased risk of APTs. This survey explores the state-of-the-art to stratify and showcase the properties of supply chain-based APTs, elaborate on reported risks from such APTs, and expand on existing defense methods. This study connects academic research with industry practices to highlight a new and growing problem. It examines supply chain compromises, offers unique insight into how these exploitations occur, and equips cybersecurity practitioners with the knowledge required to design next-generation APT defense mechanisms.**

***Index Terms*—Advanced Persistent Threats, Supply Chain Attack, Defense Methods, Classification**

I. Introduction

ACCORDING to the latest Mandiant report [1], most APTs related to ransomware attacks were orchestrated via vulnerabilities in third-party supply chain software. The Worldwide Application Security Project (OWASP) whitepaper emphasizes that SCVs attributed to third-party software dependencies have been ranked as the top vulnerabilities in large language models [2]. Furthermore, the European Union Agency for Cybersecurity (ENISA) whitepaper [3] reported that twenty-four large-scale cyber-attacks during the period of 2020-2021 were due to SCVs. Among them, 50% of the attacks were attributed to APT groups. Since the exposure of the SolarWinds supply chain attack in 2020 [4], there is a growing tendency to exploit SCVs. Because of the stealthy and widespread impact of supply chain attacks, as noted by Barr-Smith et al. [5], APTs targeting the supply chain have worsened the security situation. Therefore, creating effective and robust defense products and implementing them has become an urgent priority.

APTs arising from SCVs have not been extensively studied. It is worth to mention that most supply chain attacks are 'non-targeted', distinguishing them from 'targeted' APTs. Existing studies focus primarily on general APT [6]–[10], with limited consideration for the exploitation of the supply chain. The survey referenced [11] does not address or examine supply chain attacks. In contrast, research specifically on supply chain attacks often lacks a thorough analysis of advanced exploitation techniques [5], [12]–[15]. To address this gap, we conduct this survey to explore the techniques used by these specific APTs and to evaluate the state-of-the-art defense approaches that can counter them.

We perform a comprehensive literature review that includes academic and gray literature [16]. The academic literature review aims to identify current challenges in cyber supply chain, and state-of-the-art defense methods against supply chain attacks and APT exploitation. The majority of them are from top-tier journals (IEEE TIFS, Computers & Security) and conferences (NDSS, S&P) in the fields of APT detection and supply chain-based attacks published in recent years. In contrast, the review of gray literature directs its attention towards first-hand occurrences and practical industrial scenarios, thereby providing invaluable information. This includes white papers and reports from research labs and authoritative organizations like Google and CrowdStrike that detail the processes of exploiting vulnerabilities. These materials, based on real-world attack scenarios, provide up-to-date information and valuable insights from domain experts. The methodology for selecting the cited materials is provided in Table I. The main objective is to identify the core techniques used in supply chain-based APTs to conceal their behaviors, and offer guidance for efficient and optimized defense safeguards.

The main contributions of this survey are:

- A comprehensive literature review regarding supply chain attacks and potential APT attacks based on SCVs. We point out the difference between two groups of attacks.

Zhuoran Tan, Shameem Puthiya Parambath, Christos Anagnostopoulos, and Jeremy Singer and are with the School of Computing Science, University of Glasgow, Glasgow, Scotland, G12 8RZ. Angelos K. Marnerides is with the KIOS Centre of Excellence and the Department of Electrical & Computer Engineering, University of Cyprus, Nicosia, Cyprus. Corresponding Author: Zhuoran Tan, e-mail: z.tan.1@research.gla.ac.uk.

TABLE I: Literature Review Methodology

Methods	Search Database	Keywords	Criteria	Year Range	Examples
Academic Literature	google scholar, openreview, IEEE Xplore, ACM, Springer	supply chain security, AI security/supply chain, APT detection	top-tier conference: core A/A*, core B	2019-2024	NDSS, USENIX Security, S&P, CCS, AsiaCCS
			high-impact journals: Q1, Q2		IEEE TIFS, Computer & Security
Grey Literature	company/institution whitepapers/reports, medium	(advanced) supply chain vulnerabilities/attacks, supply chain security trend, AI supply chain	authoritative security companies, national institutions		companies: Google (Mandiant), Intel, Crowdstrike, Symantec, Mitre, Kaspersky
					institutions: OWASP, NIST, OpenSSF, ENISA

We firstly dive into the specific techniques applied by them via investigating authentic APTs based on SCVs, which has been not systematically explored before.

- A taxonomy of supply chain threats, combining insights from industry, academia, and government standards. The scope of threats covers software, hardware, dependencies, and Artificial Intelligence (AI) components. We originally cover machine learning (ML) models and used data as components of supply chain, which expands the software concept and integrates the latest practice around ML supply chain security.
- A taxonomy of the most recent defense methodologies against APT attacks and supply chain attacks. The defense methodologies have been classified into three main categories including; (i) detection methods, (ii) censoring methods, and (iii) proactive methods, in which there exist additional triage.
- Differential analysis of technique used by general APTs compared to those based on SCVs. We initially differentiate the technique stacks in two APT groups. The analysis reveals a significant difference in applied technique stacks, indicting a tailored defence mechanism for SCVs-based APTs.
- Outlining future research directions regarding the detection of APTs based on SCVs. We denote the lack of research focusing on this specific type of APT by delving into the coverage of current defence technologies. We also points out the drawbacks and limitations of those defence methods to further instruct more advanced detection research for APT based on SCVs.

The remainder of this paper is structured as follows: Section II provides the definition of APT and how it can be illustrated in the Cyber Kill Chain with the common techniques involved. Section III deals with the components involved in cyber supply chain and the triage regarding supply chain compromises. In Section IV, we compare APTs with traditional supply chain attacks, perform statistical analysis for applied techniques, propose the threat model, and provide comprehensive enumeration of attack vectors. Section V discusses the defence methodologies for these attacks and further classifies the different approaches. Section VI presents a discussion of current challenges and additional research opportunities to efficiently detect these attacks. Finally, Section VII summarizes and concludes the paper. A graphic structure illustrating the organization of this survey is provided in Fig. 1.

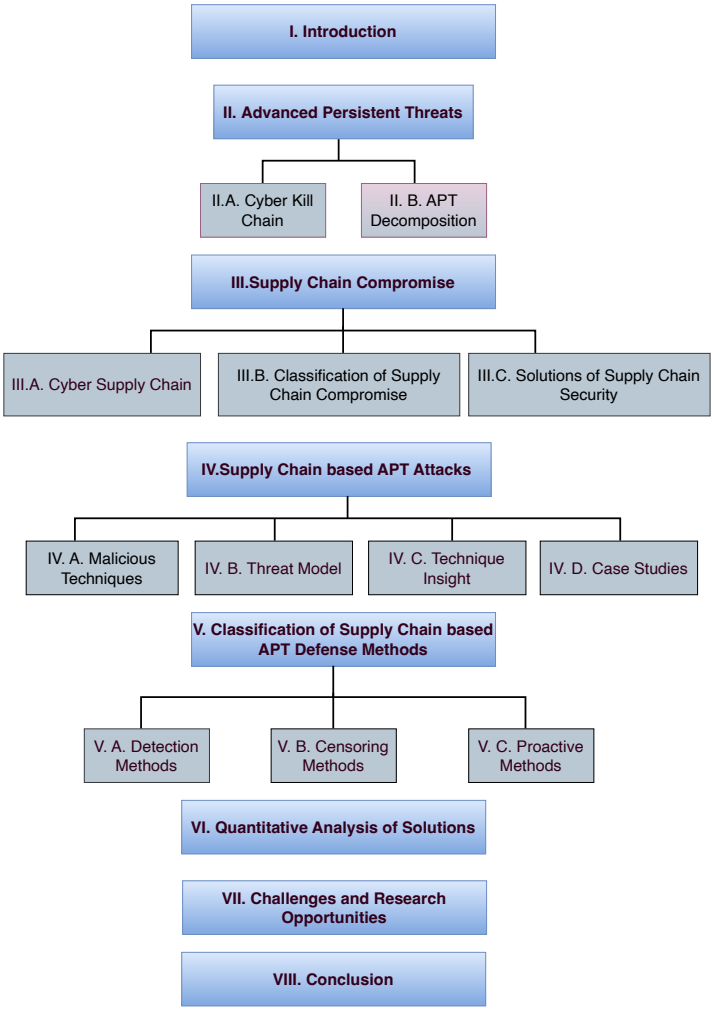

Fig. 1: Structure of Survey

II. Advanced Persistent Threats

We begin with the concept of the Cyber Kill Chain (CKC) to illustrate the generic steps of any attack vector. Subsequently, we discuss APTs by synthesizing the tailored stages of the Cyber Kill Chain. Finally, we explore a more detailed description of the common techniques and stages exploited during APT attacks.

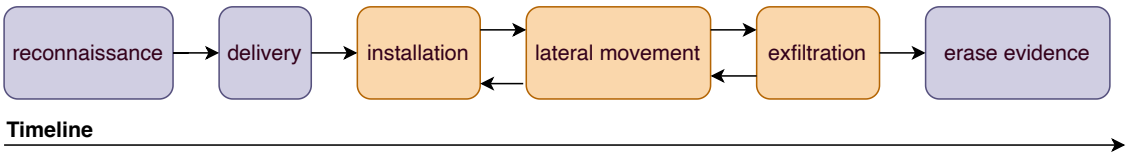

Fig. 2: APT Stages.

A. Cyber Kill Chain

The concept of the 'Cyber Kill Chain' originated as a military concept, introduced by Lockheed Martin [1] in 2011. The CKC points to a series of steps that trace the stages of a cyberattack from the early stages of reconnaissance to the exfiltration of data. The typical stages include target identification, weapon delivery, attack initiation, and ultimately target destruction or exfiltration, which also motivates understanding of APTs. Over the years, this concept has been extensively applied in information security and forms the basis for the widely recognized MITRE matrix. In addition, its content has been enriched for other variants, such as the Unified Kill Chain (UKC) [17].

The UKC offers significant improvements over these scope limitations of the CKC and the time-agnostic nature of the tactics in the MITRE matrix, which has been considered as the most comprehensive and popular version for distinguishing APT stages. The UKC model is organized around three main objectives: "In", "Through", and "Out", each containing various sub-stages or sub-tactics. For instance, the "In" objective includes tactics such as reconnaissance, resource development, delivery, exploitation, and persistence. In the following section, we refer to the core part of the UKC model to discuss the stages of APTs.

B. APT Decomposition

An APT is a sophisticated and targeted long-term cyber attack orchestrated by well-resourced and highly skilled adversaries [18]. Originally from a military context, the APT concept has expanded into the realm of the daily security society of information technology [18]. These attacks were initially directed at nation states and related entities, but have since expanded their targets to include private and organizational institutions as well [19]. Defending against APT attacks has become increasingly difficult due to their unique characteristics, including long-term latency, uncertain strategies, and strong evasion capabilities [20]. Consequently, cyber-analysts often face limited insights and a scarcity of data when investigating such attacks. Fig. 2 illustrates the typical stages of an APT attack, which include reconnaissance, delivery, installation, lateral movement, exfiltration, and erase evidence. These stages essentially work as a chain and, respectively, rely on the previous adjacent stage.

In order to explicitly illustrate the meaning and context of each stage, we refer to tactics defined in the MITRE Adversarial Tactics, Techniques, and Common Knowledge (MITRE ATT&CK) framework[2]. To our knowledge, MITRE provides a more comprehensive definition of techniques, tactics, and procedures (TTPs) for various types of attacks. In addition, it is a widely used framework for defense purposes in the industry, including Security Operations Centers (SOCs), Incident Response, Attack Simulation, and more.

Reconnaissance: An APT attack is a targeted assault aimed at gathering detailed information about a target organization's network, employees, and topology to identify potential attack vectors. The two main methodologies used are active and passive information gathering. Active gathering involves direct interaction with the target network through scans and probes, often generating logs or alerts. Passive gathering relies on open-source intelligence (OSINT), network traffic listening, sniffing, and phishing, leaving no obvious traces. According to MITRE, this reconnaissance stage may involve preparing malicious tools, malware, and resources for further tactics like initial access, Command and Control (C2), and defense evasion, essential for executing and sustaining the attack. During this stage, the exploitation of SCVs may involve gathering information about an organization's suppliers, contractors, or third-party service providers. By exploiting weak security practices in there suppliers [21], attackers can learn about targets' internal systems or workflows.

Delivery: This stage involves sending the attackers' 'weapons' to the targeted system. These weapons consist of remote access malware or malicious code [18], also known as payloads, which can be used in later stages to control the compromised target or conduct further exploitation. The delivery of these malicious payloads typically occurs through commonly used mediums, such as email attachments, compromised websites, or USB drives. During this stage, the attackers convert the information and vulnerabilities gathered in the previous phase of information gathering into a malicious payload. This conversion process is essential for achieving the initial foothold within the target system, referred to as 'initial access' in the MITRE framework. For SCVs exploitation, adversary can use compromised updates or software from a trusted supplier to deliver malicious payloads into the target organization's network. This could involve tampered legitimate software, where a routine update process becomes a method of delivering malware [4].

Installation: In diverse delivery scenarios, the installation process of malicious payloads occurs following actions such as clicking malicious links in email attachments or visiting compromised websites. In more sophisticated cases, adversaries may employ trigger mechanisms based on rare events, a typical sequence of events, or scheduling tasks [22]. This

[1]https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

[2]https://attack.mitre.org/

stage aligns with tactics specified in MITRE, encompassing execution, persistence, and privilege escalation exploitation. The first tactic, execution, involves executing malicious code to gain Unauthorized Access (UA) from the target system. Subsequently, the persistence tactic comes into play, allowing the download of malicious tools or software, such as malware, to maintain persistent control even after system reboots or the implementation of security measures. In many cases, the attacker strives to establish a Hidden Backdoor (HB) within the target organization's system to achieve persistence objectives. In a supply chain scenario, after successfully exploiting vulnerabilities, attackers can install backdoors into systems using trusted vendor tools or remote access methods, enabling them to bypass security mechanisms [23].

Lateral Movement: After infecting the device of end-user(s) within the targeted institution, the attackers engage in lateral movement, which involves expanding their reach by gaining control over other machines or devices that may hold valuable structural information or sensitive data. This stage is almost impossible to achieve with malware that has initially been uploaded or malicious scripts executed. The reason is that the restrictions imposed by access control lists or least privilege policies, the initially compromised account often lacks sufficient privileges. Consequently, lateral movement stage often involves several rounds of backward new tools or scripts upload ('installation'), and then trail exploitation for privilege escalations until compromising the credential with higher privilege. This phase is essential for expanding the attack's reach and accessing high-value assets and sensitive information within the organization. Once inside, attackers can also leverage supply chain relationships to move laterally within the network, exploiting the trust associated with vendor accounts or weaknesses in software and devices supplied by third-party vendors [24].

Exfiltration: Various APT attacks have distinct objectives, but they share a common goal: Data Exfiltration (DE). In this context, adversaries aim to collect sensitive or valuable information from compromised systems or network resources. The data they target may include financial data, Document Theft (DT), Intellectual Property Theft, personal data (ID), and credit card information, among others.

However, data stored at different locations are normally assigned with various access privileges, which requires users with various privileges to access them. The interactions with the previous stage ('lateral movement') for the higher-privilege user compromise explains this situation. Once the attackers have collected the desired information, their next step involves transferring it to controlled external servers. This transfer is accomplished through the C2 method, which involves the communication channels used to issue commands, exfiltrate data, and receive additional instructions from the attackers. Through these channels, stolen data are sent to designated external servers, providing attackers with access to exfiltrated information for malicious purposes.

In addition, once attackers obtain sensitive data, they may also use these compromised supply chain channels for exfiltration, bypassing detection. By utilizing trusted communication platforms, the exfiltration appears legitimate and avoids raising suspicion [25].

Erase Evidence: To evade detection and tracking, attackers employ various methods during the post-exploitation phase. They may choose to Delete Logs (DLS) or Remove Malware Artifacts (RMAs) generated by their activities, or uninstall malware once it is no longer needed. This stage underscores the necessity for proactive real-time detection during the early phases of an APT to reduce potential damage. The removal of logs and malware artifacts is a key reason why tracking APT activities after the full cycle is completed becomes significantly challenging. These manipulations often overlap with the 'impact' tactic defined in MITRE, which involves actions aimed at disrupting or damaging the target environment.Therefore, containing the impact at this stage is crucial to minimizing the overall damage caused by the APT attack.

Despite certain common processes shared by APTs, the techniques they employ vary significantly. This diversity in techniques, especially in complex systems, often exposes more attack surfaces, increasing susceptibility to exploitation. A notable example of this is the exploitation of Cyber Supply Chain (CSC), which has become prevalent for APT actors [5], [12], [26]. To address this, we conduct a comprehensive investigation into supply chain compromises, detailing the techniques used, and discussing proposed solutions to mitigate this vulnerability.

III. Supply Chain Compromise

The supply chain comprises a sequence of distinct products or services defined between multiple institutions or vendors. It encompasses the mechanisms of product or service delivery. Due to its diverse nature, it poses challenges in auditing, investigation, and integration. Each intermediate stage or element within the supply chain could potentially be targeted by adversaries before reaching the final consumer. As a result, supply chain-based APT attacks have increased in recent years and have become very impactful threats. Such attacks typically possess excellent levels of concealment and persistence. To understand the occurrence of supply chain attacks, it is essential to understand the components involved in the supply chain, followed by the triage of attacks targeting those components.

A. Cyber Supply Chain

The concept of supply chains encompasses the combination of organizations, individuals, resources, and technology involved in the production, distribution, and delivery of services to end users [3], [27]. Ludvigsen et al. [28] highlighted the entire supply chain lifecycle, from product and service sources to integration through design, development, manufacturing, and delivery.

In this survey, we emphasize the CSC, which is a subset of the overall supply chain. The CSC specifically deals with the digital network of organizations and systems involved in the creation, distribution, and management of digital products, services, and components. [26]. The CSC spans all stages of the lifecycle, from the initial development of hardware and software to their deployment, operation, and eventual decommissioning. It encompasses components such as hardware,

software, third-party service providers (e.g., cloud), logistics and distribution channels, and manufacturing processes. The risks associated with these components vary significantly depending on their nature.

a) Software and Hardware

The Software Supply Chain (SSC) includes both open-source and closed-source software. The primary risks involve vulnerabilities in the software or malicious code introduced during development or updates, which can compromise the entire supply chain. Additionally, the extensive dependencies in software ecosystems present potential exploitation vectors [16]. In contrast, pertains to specific devices such as firmware, servers, routers, and IoT devices. Risks in this domain often stem from compromised or counterfeit components, posing a significant threat to system integrity [29], [30].

b) Manufacturing Processes

Recently, as more emphasis has been placed on security in Software Development Lifecycle (SDLC) and Development and Operations (DevOps), the Development, Security, and Operations (DevSecOps) paradigm has been proposed to cover the practical postures to ensure the safety of software development. Human-related factors, including responsibility, transparency, and communication, also play a critical role in enhancing security throughout manufacturing processes [31].

An emerging category of risk arises from the adoption of AI technologies, referred to as Machine Learning, Security, and Operations (MLSecOps) [32], [33], which involves risks related to data, models, code, and model storage platforms [34], [35]. Additionally, privacy concerns are a prominent issue within this domain [36].

c) Logistics and Distribution Channels

This component encompasses entities responsible for the transportation and delivery of hardware or software. This phase is susceptible to risks such as tampering or theft during transit [29]. Furthermore, the integration of internet-based technologies in logistics increases vulnerabilities to unauthorized access and data breaches. The advent of quantum computing introduces potential encryption risks, further complicating supply chain security [37].

d) Third-Party Providers

Third-party providers, such as companies offering cloud computing, storage, or outsourced IT services, are integral to modern CSCs. These vendors often have access to critical systems and data, making them high-value targets for attackers. Weaknesses in these trusted providers can be exploited as footholds to deliver malware or facilitate lateral movement within target systems [21], [24].

The aforementioned perspectives mention comprehensive risks, attacks, and threats targeting CSC. They motivate us to have a dedicated triage for those threats and to provide guidance when checking CSC security in different scenarios.

B. Classification of Supply Chain Compromises

The classification of supply chain attacks is a critical step in implementing effective defense strategies and detection methods. Organizations such as ENISA [3] and NIST [38] have developed classification methodologies. Academic studies, such as [13] and [22], have conducted extensive enumerations of attack vectors for software and hardware, providing valuable information for detailed classification. In the realm of industrial work, two widely recognized frameworks, the MITRE and the SLSA frameworks [16], offer analyses of categorical supply chain threats. The MITRE framework emphasizes the objects that supply chain attacks target, such as software, hardware, and infrastructure, while the SLSA framework highlights divisions based on supply chain life cycle, encompassing the source, build, and package stages from producer to customer. Notably, hardware components and dependencies, such as chips, sensors, and firmware, are critical targets in the source and build stages, make their inclusion crucial to a holistic classification.

In addition to traditional supply chain compromise, MITRE offers a framework called the ATLAS matrix [39], covering threats to the AI system. The ATLAS matrix categorizes ML related threats within a cyber attack chain. ATLAS includes the latest academic and industrial practices to thoroughly enumerate attack vectors, specifically focusing on describing ML-related compromises. Similarly, these concepts can be extended to threats targeting hardware-based AI components, such as specialized processors, edge devices, and embedded systems, which are susceptible to tampering, reverse engineering, and side-channel attacks [30], [40].

To cover the broader landscape, it is also critical to consider the role of third-party providers and distribution tunnels in the CSC. Third-party providers, such as cloud service platforms, outsourcing vendors, and hardware manufacturers, introduce significant risks due to their integration with critical systems. Compromises in third-party services can lead to widespread vulnerabilities. Distribution tunnels, including logistics chains and digital delivery systems, present another layer of vulnerability, as attackers can intercept physical shipments of hardware or tamper with software updates during digital transfer.

We synthesize the aforementioned classification methods and generalize supply chain compromises into source, build, transform, and usage-based threats, as shown in Fig. 3. Within each category, we further expand the scope to include software, hardware, third-party dependencies, and distribution channels. For example, source-based threats include counterfeit components and malicious dependencies, build threats encompass compromised build pipelines and unauthorized access to manufacturing environments, and transform threats address malicious alterations during transport or firmware updates. Finally, usage threats encompass both runtime exploits and attacks on integrated systems, such as IoT devices or industrial control systems.

This expanded classification considers both software and hardware, focusing on the software development life cycle while addressing critical risks to hardware and dependencies in other supply chain components. We then provide a comprehensive discussion for each classified group, emphasizing the interplay between software and non-software threats.

1) Source Compromise

This compromise originates before the development or transformation phase, suggesting a greater chance of insider

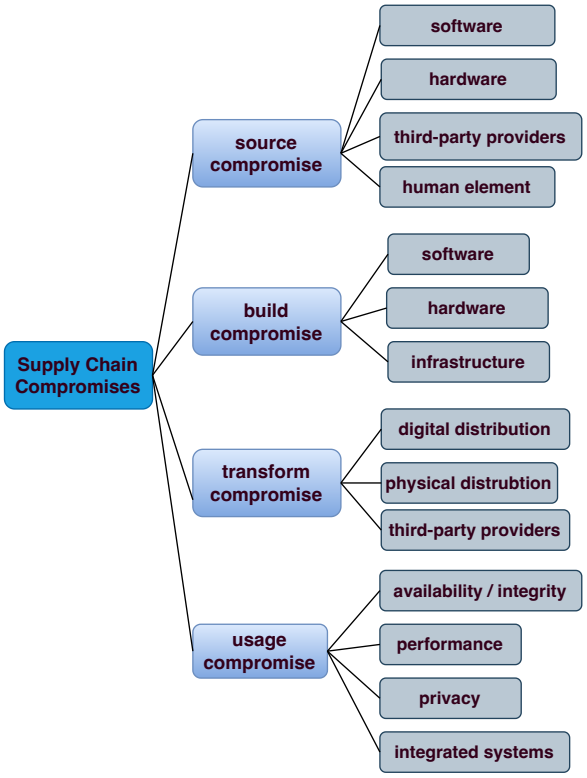

Fig. 3: Classification of Supply Chain Compromises.

threats. These threats target the initial sources of the supply chain, such as repositories, codebases, hardware components, and datasets, as well as potential human viewpoints.

This category can be divided into four subgroups, covering software, hardware, third-party providers, and human element. Software-based threats include infected packages, libraries, repositories, or malicious datasets/models. For instance, malicious updates have been observed in programming language ecosystems [41]–[43], such as Java [13], and pre-trained models on platforms like Hugging Face [44].

Hardware threats encompass counterfeit chips, tampered components, or firmware vulnerabilities. Moghadasi et al. [40] highlighted risks such as Trojan attacks and counterfeit parts, while Craciun et al. [30] described malware injections targeting signing processes.

Third-party providers may introduce risks through compromised outsourcing vendors or cloud services. Li et al. [45] pointed out the risk of the third-party risk propagation, which can lead to widespread disruptions and data breaches if not properly managed, like recent CrowdStrike blue screen accident [46]. Nata. et al. [47] also pointed out the third-party providers may pose risks related to data breaches or unauthorized access to sensitive pharmaceutical data.

Human-related threats involve insider threats from users, administrators, or supply chain manages into consideration. The compromised users often happen on code or model management platforms such as Github or Hugging face [3]. Commits from contributors with malicious intentions, especially without proper validation, have also been acted as one of the primary attacks [14], [44]. The compromise between managers or developers of these repositories or pre-trained models has become another method to achieve the exploitation of the supply chain [44].

2) Build Compromise

The build compromise denotes threats that compromise the development or assembly phase, targeting the processes, tools, or infrastructure involved in producing software or hardware. During build process, the SCVs can be classified into software, hardware, and infrastructure based compromise.

The software based build compromise can be compromised build pipelines, tampered build tools, or manipulated dependencies. Attackers may target the build pipeline to inject malicious code during the software build process, compromising the integrity of the final software product [48]. Vulnerabilities in build tools themselves can be exploited by attackers to gain control over the build process [48]. Dependency confusion attacks, where attackers upload malicious packages with the same name as legitimate internal packages to public repositories, can lead to the unintentional inclusion of malicious code in the software [48]. Hardware based compromise under build process shares similar exploitation vectors, like manufacturing processes, such as backdoor-ed hardware [49] at assembly plants or authorized firmware injection [50].

Infrastructure related compromise involves insecure continuous integration/continuous deployment (CI/CD) pipelines. Pan et al. [51] comprehensively described the risks in CI/CD pipelines, covering malicious code insertion, delayed script updates, single point of feature, and potential sensitive information exposure.

3) Transform Compromise

Transform compromises exploit vulnerabilities during the transmission or delivery of components, whether physical or digital. These compromises exploit the distribution and update mechanisms to insert malicious payloads or intercept sensitive data. Three sub-categories are included, which are digital distribution, physical distribution, and third-party providers.

Digital distribution threats include hijacked software updates, man-in-the-middle (MITM) attacks during data transmission. NotPetya [52], as a typical example, exploited the compromised ME-Doc software update server to insert malicious payloads into updates. Another more recent example is the well-known SolarWinds attack, in which updates in Orion Platform have been tainted by the malicious code called “SUNBURST“ [53].

Physical threats involve tempering with hardware shipments, such as chip swapping or device interception in transit [54]. Aniello et al. [55] discussed how attackers could forge intact items or tamper with them before their identifiers were computed. This allows counterfeit integrated circuits (ICs) to be introduced into the supply chain without detection, posing severe risks, especially in critical infrastructure applications.

[3]https://huggingface.co/

Third-party providers, such as logistics firms or content delivery networks (CDNs), introduce additional risks. Ghaznavi et al. [56] discussed two specific risks existing in CDNs, covering security vulnerabilities and infrastructure challenges, in which CDNs can be exploited to bypass security measures. The risks in logistic providers have been illustrated in [57], in which the operational risks like cost uncertainty, quality issues, can disrupt logistic operations and affect service quality.

4) Usage Compromise

Usage compromises occur during the operational phase, targeting deployed systems or their integrated components. These compromised affect the integrity, availability, or performance os systems in a real-world use. Four sub categories have been included under this group, which are availability/integrity, performance, privacy, and integrated systems.

Availability/integrity compromises can contain runtime exploits, data manipulation, or denial of service (DOS) attack. For examples, deletion of code and unavailable dependencies have been discussed in [58]. Additionally, adversaries may modify the original information without detection or potentially bypass existing controls in place [59], [60].

Performance-based threats can lead to resource exhaustion or degradation of system efficiency. These threats are common in the ML application in the online learning or retraining stage, especially when the adversary has the potential to poison the data used for retraining or observe the model output [61], [61].

Privacy risks involve data breaches, unauthorized access to sensitive information, or hardware-based side-channel attacks [62]. Rigaki et al. [63] conducted a survey on privacy attacks in ML with different configurations. Depending on knowledge level of adversary, the adversary may have access to the model parameters and API, which present potential attack vectors for training set and model information leakage.

Integrated systems are vulnerable to exploits targeting IoT devices and industrial control systems [64], [65]. By categorizing these threats, organizations can better understand and mitigate vulnerabilities within their supply chains, ensuring targeted and effective security measures. Numerous solutions have been proposed to address threats specifically targeting the supply chain.

C. Solutions of Supply Chain Security

Significant efforts have been dedicated to strengthening supply chain security, particularly through the application of advanced machine learning and deep learning techniques. In this section, we highlight some of the latest solutions presented in top-tier conferences and journals, providing a comparative analysis of their approaches and effectiveness.

Tran et al. [66] introduced a BERT-based architecture that utilizes the self-attention mechanism to detect Python vulnerable statements. This architecture replaced traditional Graph Neural Network (GNN) models with a flexible BERT-like architecture, allowing for the learning of complex relationships between code statements without predefined graph structures.

Lu et al. [67] proposed a novel vulnerability detection approach called GRACE, which integrates graph structural information and in-context learning to enhance large language model (LLM)-based vulnerability detection. This solution demonstrated retrieval approach that identifies relevant code examples based on semantic, lexical and syntactics similarities.

Khan et al. [68] employed a federated learning-based detection model to identify intrusions in Supply Chain 4.0 networks. This model further leveraged Gated Recurrent Units (GRUs) to manage data retention and forgetting. However, the generality ability still required evaluation on live data from supply chain networks of industrial cyber-physical systems (CPS).

Natampalli et al. [69] proposed a four-layered multi-tiered architecture, integrating IoT, Edge, Fog, and Cloud computing to enhance the pharmaceutical supply chain management (SCM) system's responsiveness and security. They unitized Blockchain technology, like Bitcoin, for secure transaction managements, ensuring temper-proof records through distributed ledgers and smart contracts. Another work made by Cerchione et al. [70] also employed a blockchain-based IoT architecture integrated into existing supply chain operations to enhance efficiency and transparency.

Ladisa et al. [71] employed an approach for that utilizes a set of language-independent features to detect malicious packages in npm and PyPi. The training models, which are general classification algorithms,like Decision Tree, were capable of identifying malicious packages by capturing commonalities between two ecosystems.

Liang et al. [72] presented a novel approach for detecting mobile applications using advanced machine learning techniques. The method created graph representation of code property graphs (CPGs) from source code and vectorized nodes in graph with CodeBERT. The detection model combined adaptive attention of Transformers with Graph Convolutional Networks (GCN).

Rozi et al. [73] proposed enhanced CPG representation to improve vulnerability detection in source code. This representation incorporated additional information about variable names and data types. This approach, combing graph-based analysis, addresses limitations of existing methods by capturing semantic meaning and contextual factors.

Dai et al. [74] proposed dynamic transfer learning techniques called Kernelized Cross-Project (KCC) to enhance cross-project just-in-time software defect prediction (JIT-SDP). The proposed method utilized the Kernel Variance Matching (KVM) to adapt marginal probability distributions between source and target projects. It also combined CatBoost algorithm, due to its robustness to outliers and missing values, to construct JIT-SDP.

Wang et al. [75] presented a semi-automated approach that integrated horizontal and vertical traceability to identify dependencies among high-level security requirements (HSRs) and low-level security requirements (LSRs). This approach also utilized relevance feedback (RF) to enhance automated requirements tracing by incorporating indicator terms.

Hu et al. [76] proposed a deep reinforcement learning algorithm known as DQN, which is combined with LightGBM classifiers to form the DQN-LightGBM model for improved mining performance. Additionally, an attention mechanism was also introduced to enhance feature extraction accuracy.

Since different datasets are used in these solutions, directly comparing their specific detection performance against a common baseline is not feasible. However, we introduce several shared metrics such as scalability, generality, and computational overhead to enable a level-ground comparison. The generality parameter denotes whether the solutions can adapt to new data and potential unknown threats. Table VIII presents the results of this comparative analysis.

The table shows that most recent efforts on supply chain defence concentrate on source code-based detection, often integrating graph-based techniques or deep learning methods. The scalability and generality vary from diverse technology choices. For secure and private transformations, Blockchain remains the preferred solution. It also provides general better scalability and generality. Dependency checks typically utilize dependency analysis or specialized machine learning approaches, such as reinforcement learning.

Although solutions for supply chain security have achieved significant progress and demonstrated real-world effectiveness, APT-based supply chain vulnerabilities often follow narrower exploitation paths. This specificity necessitates the development of tailored and adaptive countermeasures.

IV. Supply Chain based APT Attacks

An attack that qualifies as an APT possesses distinct characteristics. According to [19], three primary attributes differentiate advanced attacks from common attacks: the effort taken to prevent it, the requirement of high adaptation, and the display of novelty in attack variants. This framework can help identify advanced threats in the supply chain compared to common supply chain exploitations.

A Supply Chain Attack (SCA) is a key element of supply chain-based APTs. However, an SCA usually does not target specific entities. The software or hardware targeted [41], [77] is absolutely dependent on its value or popularity, gauged by its number of downloads. More downloads increase the magnitude of potential downstream impacts.

Using the example of an OSS attack, this type of attack spans stages from code contribution to package distribution [16], effectively encompassing the entire software development cycle. However, software exploitation typically serves as the initial stage for APTs, enabling unauthorized access or establishing a foothold for further exploitations.

In the following sections, we demonstrate specific techniques inside highly stealthy APTs and in more targeted SCVs-based APTs, culminating in a threat model derived through differential analysis. We also explore the insights of the techniques to illustrate the characteristics of SCVs-based APTs

A. Malicious Techniques

We categorize the applied techniques into two groups of APTs based on detailed exploitation analysis reports. The first group, detailed in Table VII, focuses on highly stealthy APT activities. In this table, the `techniques` column lists primary techniques from whitepapers, `initial stage` describes techniques used to establish an initial foothold, and `communication` covers methods for maintaining hidden communication with controlled servers or sites to evade detection systems. Additionally, the `supply chain` column indicates whether supply chain exploitation has been used by the APT groups, including historically applied techniques. The goal is to provide information on how APT groups achieve high levels of stealth and the extent to which they exploit supply chains, helping to assess the risk of these exploitable tendencies in APT strategies.

The second group presented in Table II is tailored to the exploitation of the supply chain. We have categorized extracted techniques into different stages in order to disclose the diverse technique stacks used in various stages. Missing values result from limited information disclosed in whitepapers.

During "Installation" stage, Malware may utilize file system timestamps to verify that the product has been deployed for several weeks before initiating its first beacon, which effectively evades detection mechanisms like sandboxes or other monitoring tools [78]. This time-based technique has also been observed during the delivery of second-stage payloads in attacks like the CCleaner incident [79]. Another common approach involves relying on the user execution of signed but trojanized installation packages to activate the malware [86]. Additionally, exploiting software updates, such as tampering with MSI files during legitimate ASUS live updates, is another effective method. Default permissions on certain installation directories may allow arbitrary file execution under these circumstances [87].

In "Lateral Movement" stage, attackers often exploit local valid accounts or leaked credentials information to facilitate lateral movement. For example, SolarWinds has been observed using stolen NTLM hashes via tools like Mimikatz to perform this type of movement without requiring plaintext credentials [88]. Moreover, legitimate software, such as remote administration tools or system utilities, may also be exploited for lateral movement. NotPetya, for instance, utilized the EternalBlue SMBv1 exploit and Windows Admin Shares to spread across networks [81]. The Kaseya VSA attack leveraged DLL side-loading to execute unauthorized commands by exploiting legitimate software. Similarly, HAFNIUM targeted SMB shares and administrative shares to execute commands on compromised systems [85].

A comparison between the techniques used in general APTs with those based on SCVs reveals significant divergence. One major distinction lies the techniques in `Initial Stage` of Table VII significantly differ from what covered in `Foothold Establishment` of Table II.

This representation of techniques in two groups and the comparison of them motivate us to propose a custom threat model for APTs based on SCVs and to outline a custom defense mechanism that differs from general APT activities.

B. Threat Model

Recalling the discussion on supply chain compromise, it can be inferred that SSC attacks remain the primary method of exploitation in this context [5]. This is because software, particularly OSS, is more accessible to a wide range of users,

TABLE II: Supply Chain Based APT Attacks

Name	Foothold Establishment	Delivery	Installation	Lateral Movement	Exfiltration	Erase Evidence
SolarWinds [78]	Drive-by Compromise, Trusted Relationship, Malicious Code Injection(MCI), Compromise Update Mechanism (CUM), Compromise Build Environment (CBE)	Update	Remote Services, Activate Backdoor	Pass the Hash, Remote Services, Web Protocols, Valid Accounts	Exfiltration Over C2 Channel	DLS, Modify System Timestamps, Obfuscate Activities
CCleaner [79]	Trusted Relationship, CUM, UA, MCI, CBE	Update	Upon Download/Execute	N/A	Exfiltration Over C2 Channel	N/A
ASUS [80]	UA, Trusted Relationship	Software Package, Update	Initial Setup or Update	N/A	N/A	N/A
NotPetya [81]	HU, Trusted Relationship	Update	Exploitation for Privilege Escalation	Remote Services: SMB/Windows Admin Shares, Lateral Tool Transfer	Data Encrypted for Impact	Data Destruction
Kaseya VSA [82]	Trusted Relationship, ZDV, UA	Trusted Relationship	Automatic	Update	Hijack Execution Flow: DLL Side-Loading	N/A
Accellion FTA [83]	Trusted Relationship, ZDV, UA	Web Shell	Create or Modify System Process, Create Account	N/A	Exfiltration Over Alternative Protocol	N/A
Codecov [84]	Drive-by Compromise, UA	Bash Uploader	Backdoor	N/A	N/A	N/A
HAFNIUM [85]	Trusted Relationship, Exploit Public-Facing Application, ZDV, UA	Web Shell	Create or Modify System Process, Create Account, Command and Scripting Interpreter: Windows Command Shell	OS Credential Dumping: Security Account Manager, Remote Services: SMB/Windows Admin Shares	Exfiltration Over Web Service: Aspera Faspex	N/A

leading to broader attack vectors. In contrast, Hardware Supply Chain (HSC) attacks involve a significant diversity of devices and infrastructures [22], [30], [89], making it difficult to develop a single threat model that covers all HSC scenarios.

To ensure that the proposed threat model gains widespread acceptance, we focus specifically on SSC [13], [16]. We integrate techniques from Table II to develop the threat model shown in Fig. 4. Additionally, we streamline the sources by concentrating on code repertoires and software, instead of enumerating all potential types.

Within the threat model, injecting malicious code acts as the initial exploitation aimed at gaining unauthorized access while maintaining concealment [16]. Injection relies on user actions to introduce the code into the target system. Various scenarios enable this, such as library dependencies, updates, and downloads [5], [13], [77], [90]. For example, users might unintentionally add infected versions of Python libraries as dependencies, or unverified updates might automatically install malicious code. Furthermore, downloading from unofficial forks can also introduce malicious commits and unpatched vulnerabilities from the original repositories [90].

After malicious code is inserted into a user's system, diversely designed trigger mechanisms can activate it. These triggers can start automatically upon downloads or updates. More sophisticated exploitation involves setting up specific events based on scheduling tasks, rare occurrences, or predetermined periods of concealment [4], [22]. Subsequent exploitation follows the general stages of APT activities. Triggered behaviors may involve downloading second-stage malware with additional functions [91] or establishing contact with additional services such as C2 servers. Through further exploitation, the adversary may engage in reconnaissance once again to gather more internal information or conduct internal phishing. The ultimate goal is typically to acquire more credential information.

Once malicious actors gain access to valid accounts, they expand their influence through the lateral movement stage to infect more devices. By obtaining administrative privilege through an admin account, the final goal is to exfiltrate data to their C2 server. To avoid detection during this process, adversaries send data in batches over multiple days and erase any traces of their activities. Therefore, the ultimate intention of supply chain-based APT attacks is not to compromise integrity or cause immediate damage. The widespread adoption of supply chain attacks increases the likelihood of gaining unauthorized access to the targeted systems.

In this section, we present a typical threat model that focuses on common usage scenarios. In particular, the threat model does not include general reconnaissance in its initial stage, although the adversary may still investigate information against targets in reality. However, this omission encourages us to investigate whether APTs based on SCVs have customized exploitation methods within the kill chain. This investigation is crucial to identify specific APT groups and develop effective mitigation strategies.

C. Technique Insight

To efficiently identify and mitigate threats, one practical approach is to identify the specific APT group [92] based on traces from critical stages. These critical stages are defined by the techniques most commonly used in them. For detection, it is more effective to focus on these critical stages first, then extend to earlier or later stages.

There is a trade-off between detection accuracy and computational resource constraints, necessitating an optimal defense mechanism in practical scenarios [20]. Although a holistic chain describes the APT stages, it is crucial to assess the data available from each stage based on the characteristics of APT in the supply chain context.

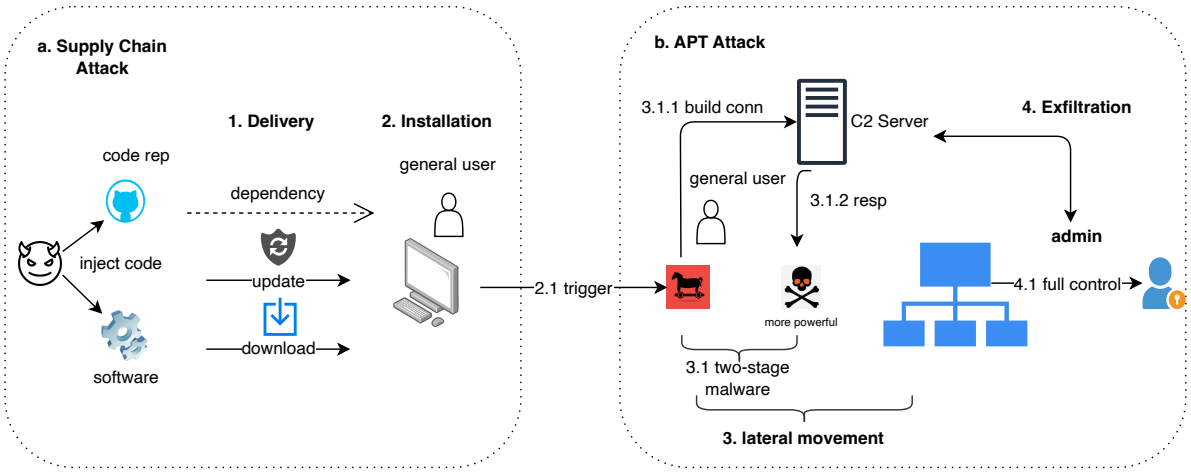

Fig. 4: Threat Model of Supply Chain based APT. **Supply Chain Exploitation:** 1. Delivery the weapon with potential three primary approaches (dashed line represents the indirect way); 2. End users install infected tools and vulnerabilities get triggered upon installation; **APT Stage**: 3. Lateral Movement through download of multi-stage malware to compromise admin; 4. Exfiltration to export critical data from compromised systems to attacker controlled server

Additionally, to distinguish the type of APT based on SCVs, it is essential to understand how they differ from general APTs. The most efficient approach is to observe the differences in techniques used at various stages. Once these differences are identified, the implementation of effective mitigation solutions becomes more feasible [93], [94].

To achieve this goal, we start with a stacked vertical bar chart that visualizes all the techniques used in various tactics, including the shared techniques between two groups, as detailed in Table VII and Table II. As illustrated in Fig. 5, we use pyattack [4] to associate the name of the techniques with the corresponding tactics in the MITRE matrix and count the number of unique techniques for each tactic. Assuming that the techniques listed in Table VII are T_1 and those in Table II are T_2, the blue section of the bar represents all unique techniques found in either group, denoted as $T\_1 \cup T\_2$. This illustrates the variety of techniques used across different tactics. In contrast, the orange section represents the techniques that are common to both groups, denoted as $T\_1 \cap T\_2$. This chart reveals that most techniques are concentrated within three tactics: `persistence`, `privilege escalation`, and `defense evasion`. The common shared techniques make up a small portion of the total set of techniques. Furthermore, there is no overlap in later tactics after `defense evasion`. This indicates a significant difference in the technique stacks between the two groups.

To demonstrate how different the technique stacks are, we provide a vertical bar chart shown in Fig. 6 with the same matching method. This orange section represents the total of unique techniques applied in APTs based on SCVs, which are retrieved as technique t where $\{t \in T2 \mid t \notin T1\}$. The blue section shows the total number of unique techniques in both sets combined. Compared with Fig. 6, we find that the previously non-overlapping tactics after `defense evasion` in Fig. 6 have techniques that are unique to the supply chain,

[4]https://github.com/swimlane/pyattck

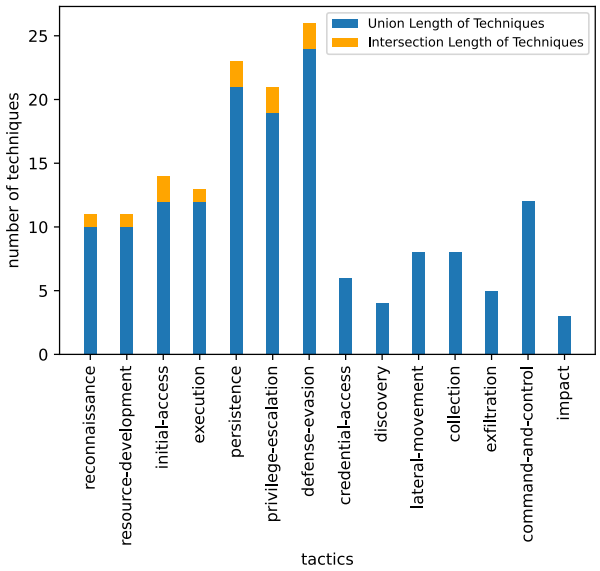

Fig. 5: Comparison of Union and Intersection Techniques in Tactics between all APTs

even though there are no unique techniques that have been found in `discovery` and `collection`. This observation confirms the distinctiveness of these tactics in the context of supply chain-based exploitation. Furthermore, while there are some overlaps in other tactics between the two groups, the techniques that exist exclusively in the supply chain still exhibit relatively large numbers. In conclusion, the technique stacks utilized by general APTs and APTs based on the supply chain show significant dissimilarities.

To provide a more detailed analysis of APT techniques leveraging supply chain vulnerabilities, we conduct two case studies: one focusing on software-based exploitation and the other on hardware-oriented exploitation.

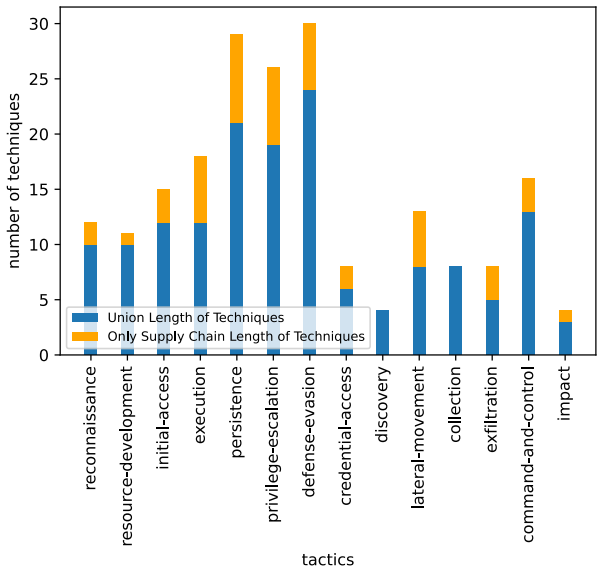

Fig. 6: Comparison of Union Techniques in all APTs and Techniques only in APT based on Supply Chain

D. Case Studies

In this section, we present two case studies focusing on HSC and SSC, respectively, to illustrate nearly real-world examples of exploitation in supply chains. These case studies shed light on the vulnerabilities and potential threats within both types of supply chains, providing valuable information on the challenges organizations face in securing their supply chain processes.

1) SolarWinds

The SolarWinds supply chain attack stands out as a prominent example of exploitation within SSC and has been extensively studied both in academia and in the cybersecurity industry [88], [95]. Numerous research efforts have contributed to understanding the intricacies of the attack and devising effective countermeasures.

Alkhadra et al. [95] offered a general analysis of the infection process and affected sections, but did not take a detailed technical look at the subject. In contrast, Martinez et al. [96] presented an in-depth case study, which incorporates the cyber kill chain and provides a comprehensive attack timeline. The case study conducted by Barr-Smith et al. [5] focuses on the SolarWinds attack's trojans, Sunburst and Supernova. Their analysis seeks potential attack clues and traces to aid detection, particularly for advanced hidden techniques employed.

Building on previous work, we abstract and model the attack flow, as illustrated in Fig. 9, which depicts Sunburst, a trojanized version of the SolarWinds Orion plug-in. This malware includes a backdoor and communicates with third-party servers via HTTP. Developed during resource development stage, the malware employs a virtual private server to provide multiple IP addresses, aiming to evade detection. The choice of IP addresses from the victim's country further complicates detection. However, unusual IP access patterns, such as rapid travel rates or logins from unexpected Autonomous System Numbers, indicate potential malicious activity.

The malware operates by retrieving and executing commands after a dormant period of up to two weeks. To disguise network traffic as legitimate protocol, communication with the extra server is employed. Reconnaissance data is stored within the plugin configuration file. Lateral movement is facilitated using valid credentials, and graphed logon activities can reveal one-to-many relationships indicative of malicious access.

To evade detection during payload execution, the adversary employs temporary file replacement and temporary task modification techniques. Detecting such behaviors involves examining logs for SMB sessions, where a delete-create-execute-delete-create pattern may be observed in a short time. In-depth malware analysis, detection of Domain Generation Algorithm (DGA) usage, and disguised network traffic detection are also potential methods to uncover exploitation. While these detection methods may introduce bias, proper correlation can help reduce false positives.

2) Nuclear Power

As a typical case of hardware-based supply chain attack, the nuclear power case study is derived from information on the well-known hardware-based APT attack Stuxnet [97]. This case is used to illustrate a potential attack scenario in HSC.

Stuxnet was a targeted threat specifically aimed at industrial control systems (ICS), such as gas pipelines and power plants. Its objective was to reprogram ICS by modifying code on Programmable Logic Controllers (PLCs). According to the report [98], Stuxnet utilized multiple zero-day exploits, a Windows rootkit, anti-virus evasion techniques, and C2 mechanisms. The most likely initial access point was through infecting a willing or unknown third party, making it relevant for our HSC scenario. The infection technique primarily involved DLL injection. Additionally, we refer to existing research on hardware supply chain attacks in nuclear power infrastructures [22], [99] to develop a more explicit attack scenario.

In the provided diagram (Fig. 8), we present the entire life cycle of hardware development, including integration into end-user systems. The connection between the PLC and the final product centrifuge can be established during the hardware integration process. However, malicious code can also be introduced at various stages, such as firmware, sub-assemblies, and system integration steps. Different stages of development may be targeted by diverse attacks. We have identified five critical threats, including (1) malicious injection or (2) substitution, (3) tool alteration , (4) configuration modification, and (5) theft of IP or data [22], [98]. These attacks can be launched at any stage, whether during specific building or testing phases, or even during the delivery process.

The reconnaissance stage in HSC mainly involves the theft of ICS schematics, where each PLC is uniquely configured. Hence, successful attacks involving data or design theft can contribute to reconnaissance in any stage. The development tool with inserted malicious code can introduce a payload during compiling or integration processes, aligning with build threats outlined in the SLSA framework. By enumerating all processes, it becomes clearer for malicious actors to identify which threats can be executed at different stages. In addition, customized defense policies can be deployed accordingly.

TABLE III: Collection of Academic Methods in Research Papers

Defense Methods	References	causality	provenance	cross-host correlate	novel features	deep learning	rule-based	differential analysis	graph inference	anomaly scoring	state-based track	semantic analysis	machine learning	risk management	program analysis	attack vector enum	block chain	history/pattern
Detection	Irshad *et al.* (2021) [9]	×	✓	✓	✓	×	×	×	×	×	×	×	×	×	×	×	×	×
	Alageel *et al.* (2021) [6]	×	×	×	✓	×	×	×	×	×	×	×	×	×	×	×	×	✓
	Ho *et al.* (2021) [101]	×	×	×	×	×	✓	×	✓	✓	×	×	×	×	×	×	×	✓
	Bhattarai *et al.* (2022) [8]	✓	×	✓	×	×	×	×	×	×	×	×	×	×	×	×	×	✓
	Yang *et al.* (2022) [102]	×	×	×	✓	×	×	×	×	×	×	×	×	×	×	×	×	×
	Xiong *et al.* (2022) [103]	×	✓	×	×	×	×	×	×	×	✓	×	×	×	×	×	×	×
	King *et al.*(2023) [104]	×	×	×	×	✓	×	×	×	×	×	×	×	×	×	×	×	✓
	Du *et al.* (2024) [105]	×	×	×	×	✓	×	×	×	×	×	✓	×	×	✓	×	×	×
	Pasquale *et al.* (2024) [106]	×	×	×	×	×	×	×	×	×	✓	✓	×	×	✓	✓	×	×
Censoring	Vasilakis *et al.* (2021) [15]	×	×	×	×	×	×	×	×	×	×	×	✓	×	×	×	×	×
	Barr-Smith *et al.* (2022) [5]	×	×	×	×	×	×	✓	×	×	×	×	×	×	×	×	×	×
	Rieger *et al.* (2022) [107]	×	×	×	×	×	×	×	×	×	×	×	✓	×	×	×	×	×
	Ladisa *et al.* (2022) [13]	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	×	×	×
	Cheng *et al.* (2023) [108]	×	×	×	×	×	×	×	×	×	×	×	✓	×	×	×	×	×
	Froh *et al.* (2023) [109]	×	×	×	×	×	×	✓	×	×	×	×	×	×	×	×	×	×
	Liu *et al.* (2024) [110]	×	×	×	×	✓	×	×	✓	×	×	✓	×	×	✓	×	×	×
Proactive	Neo C. K. Yiu (2021) [111]	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	×
	Eggers *et al.* (2021) [22]	×	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	×	×
	Jiang *et al.* (2022) [44]	×	×	×	×	×	×	×	×	×	×	×	×	✓	×	×	×	×
	Ladisa *et al.* (2022) [16]	×	×	×	×	×	×	×	×	×	×	×	×	✓	×	×	×	×
	Islam *et al.* (2022) [112]	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	×
	Ren *et al.* (2023) [113]	×	×	×	×	✓	×	✓	×	×	×	×	×	×	×	×	×	×
	Yin *et al.* (2023) [114]	×	×	×	×	✓	×	✓	×	×	×	×	×	×	×	×	×	✓

In the case of ICS, due to the strong privacy and confidentiality requirements, direct traffic monitoring may not be feasible. Therefore, we recommend implementing code-based verification before integration and adopting the compromised DevSecOps approach [100] during the building process for enhanced detection and prevention.

V. Classification of Supply Chain based APT Defense Methods

To illustrate the latest trends in defense against APTs based on SCVs, we chose scholarly articles from top-tier journals and conferences over the past four years. The comprehensive statistical analysis is presented in Table III, covering both domains. The first half of the references up to Exorcist [5] highlights the most recent defense methods in APT detection. The remaining references stress defense methods against supply chain attacks. We record the detection methods employed and whether the introduced methods are historical data or pattern-based. This dimension indicates whether a large amount of training data is required for the implementation of detection. By integrating all the papers analyzed, we have formulated a classification tree in Fig. 7 that includes all the detection methods investigated. This triage tree serves as a guide for the development and deployment of defense products against SCV-based APTs, providing valuable insights and instructions to researchers and practitioners in the field.

A. Detection Methods

This category encompasses research methods that focus on APT detection. Within this category, we have divided the corresponding research papers and methods into two subcategories. Attack Reconstruction and Anomaly Detection.

1) Attack Reconstruction

Attack reconstruction involves the application of technologies to recover attack paths or graphs by analyzing the original data sources or correlating information from alerts generated by intrusion detection systems. This process encompasses three main related technologies: provenance, causality, and correlation analysis. Using these techniques, attack reconstruction has proven to be an effective solution to mitigate possible false predictions generated in anomaly detection [8], [101].

a) Data Provenance

Data provenance refers to documentation of the origin, creation, and propagation process of data, encompassing its lineage and derivation [115]. Due to its ability to track the history and origin of information throughout its life cycle, data provenance has found significant use in the field of cybersecurity [7], [10], [103], particularly in distributed environments [116]. In the context of APT detection, data provenance can offer valuable insights into specific scenarios.

To address efficiency concerns associated with provenance analysis, [103] proposed a state-based framework called CONAN, which does not store historical data. This approach eliminates unnecessary phases of APT attacks and only retains approximately one in a thousand events in a database. As a result, it efficiently detects both known and unknown APT attacks with minimal False Positive Rate (FPR). However, debates exist regarding the framework's ability to detect APT attacks employing new or unknown techniques, its representation of all real-world scenarios, and the required computational resources for real-time processing of large data volumes.

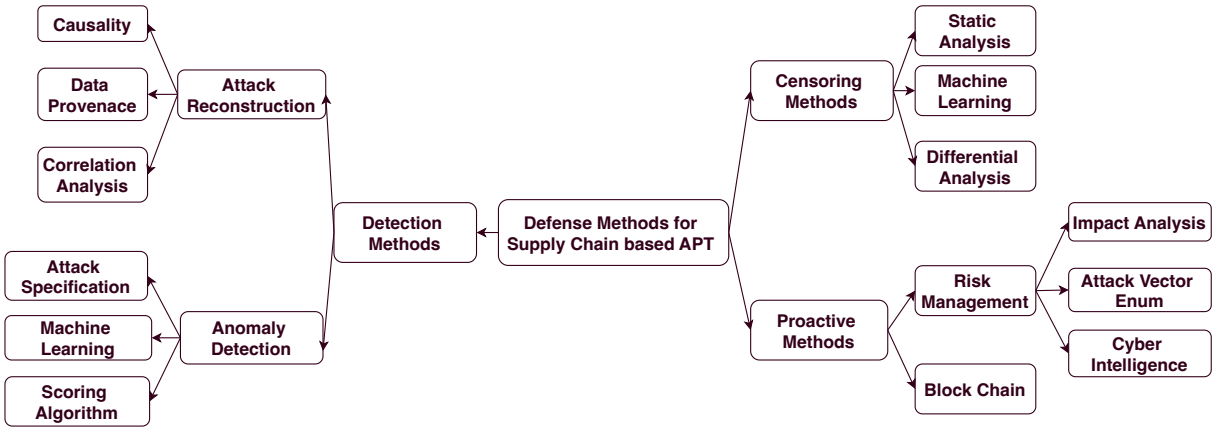

Fig. 7: Classification of Defense Methods

An attack investigation framework, ATLAS, was proposed in [7]. This system addresses the issue of alert fatigue resulting from a large number of threat alerts. It integrates natural language processing (NLP) and ML techniques for data provenance analysis to distinguish attack and non-attack sequences. By employing NLP techniques such as word embedding and lemmatization, contextual event sequences are mapped to numerical vectors. The Long-Short Term Memory (LSTM) model is then trained to identify backward and forward relationships between events within sequences. The data provenance in this system helps reconstruct the attack story by establishing causal graphs based on identified attack events. The approach outperforms the intrusion detection and association analysis methods by precisely locating critical steps and learning historical attack patterns.

The TRACE provenance tracking system, as discussed in [9], appears to be a promising approach to efficient detection of APT stage at the enterprise level. The integration of static analysis, continuous improvement efforts over several years, and the goal to construct an enterprise-wide causal graph for APT activity identification are noteworthy. It is particularly interesting that the system has undergone practical testing in adversarial engagements, achieving an 80% detection coverage for APT stages. The potential future enhancements, such as integrating an in-kernel cache-based audit logging system and model-based causality inference, indicate a commitment to ongoing improvement.

Data provenance, while offering valuable insight into the origin and propagation of data, faces challenges, as mentioned in [103]. The challenges of efficiency, memory, computation, and storage resource become more pronounced, especially in real-world long-term APT scenarios. Balancing the need for comprehensive analysis with the constraints of resource usage is a critical consideration for the practical implementation of data provenance in cybersecurity defense strategies.

b) Causality

Causality, as it relates to understanding cause-and-effect relationships between events, involves causal discovery and causal inference. Causal discovery aims to infer causal relationships between variables from observational data when no existing relationships are assumed [117]. On the other hand, causal inference quantifies the influence of one variable on others, relying on prior assumptions about their relationships [118]. These concepts, rooted in probabilistic theory and statistics, play a crucial role in understanding and modeling causal relationships in various domains, including cybersecurity where they can contribute to effective APT detection and response strategies.

Bhattarai et al. [8] presented the SteinerLog system, an automatic tool to detect and reconstruct APT attack campaigns in real time. It used the prize-collecting Steiner tree algorithm and hierarchical graph traversal to correlate cross-host attack activities. Risk scores were assigned based on the rarity and sparsity of anomaly events. The system builds attack stages using temporal traversal and incorporates historical attack patterns and customer behaviors to improve detection accuracy and reduce investigation time.

In [7], causal graphs were constructed from recognized attack events in audit logs to address alert fatigue issues. The work in [10] extracted causal dependencies within alerts generated by EDR systems to reconstruct attacks. The authors of [119] collected causality logs from browsers' execution history to detect watering hole attacks. Furthermore, the study [9] correlated the provenance of individual hosts to create an enterprise-wide causal graph to locate APTs at the enterprise level. Correlation analysis is essential for integrating information from multiple devices or hosts.

However, causality related methods face challenges similar to those in data provenance, notably the dependence explosion [8], [9], for which no efficient solution exists. The explosion introduces bias and makes the final formed graphs challenging to analyze, or consumes excessive resources, which is impractical in real detection scenarios.

c) Correlation Analysis

This analysis seeks to reveal hidden relationships in large data sets through co-occurrence [120]. It uses two key concepts: frequent item sets, which identify relationships and co-occurrences, and association rules, which define interdependence and conditional relationships. Frequent item sets measure coverage, while association rules are evaluated based

on support and confidence.

Yang et al. [102] integrated both causality and correlation to support semantic analysis to detect APTs over a long time interval from the alerts of existing systems. By mining the causality between anomalous events, the system reconstructs alert chains and utilizes Latent Dirichlet Allocation (LDA) to model the semantic context of these chains. The LDA is a hierarchical Bayesian model that allows capturing latent attack intent and reconstructing APT scenarios. The test in a real-world dataset yielded a better precision rate compared to CONAN [103].

To detect complex watering hole attacks that are often seen in the early stages of APT, Allen et al. [119] created MNEMOSYNE, a postmortem forensic analysis tool. MNEMOSYNE reconstructs, investigates, and assesses these attacks with minimal browser modification. It gathers causality logs from browser activities and uses versioning and user-level analysis to identify compromised websites and their changes. This system provides a comprehensive internal forensic investigation, outperforming low-level semantic detection solutions.

The correlation analysis increases the opportunity for precise detection of certain threats. However, the drawbacks are obvious, like the failure against attacks that do not utilize system calls or implicitly use them. Additionally, the related approaches struggled with attacks that have multiple entry points, limiting its ability to detect complex APT scenarios [121].

Although attack reconstruction technologies are prominent in the latest novel detection methods, some still rely on anomaly detection. Moreover, attack reconstruction typically filters alerts generated by anomaly detection systems.

2) Anomaly Detection

Anomaly detection is a fundamental concept employed by defenders to identify abnormal instances within original data. These anomalies can manifest as unusual data points that do not conform to the expected patterns or as outlier values that deviate significantly from the normal range. The definition of an anomaly can vary across different scenarios, thereby necessitating a diverse range of detection methods. In this study, we present a categorization of these methods, which results in the identification of three main groups.

a) Attack Specification

The attack specification is a rule-based detection method that defines specific rules based on empirical findings about various attacks, often referred to as Indicators of Compromise (IOCs) or Indicators of Attack (IOAs) in industrial contexts [8], [122].

Alageel et al. [6] introduced the HAWK-EYE system, which aims to achieve holistic detection for APT C2 domains. By generating novel features from semantic and structural properties of domains using DNS infrastructure, the system can classify domain names and evaluate the performance of various features. It serves as a parallel or first filter role for network intrusion detection systems (NIDS) and enhances the detection ability with new features.

To effectively detect an attack in lateral movement stage, Grant et al. [101] proposed the system Hopper, using common-collected log data to detect lateral movement attack with a manageable rate of false alarms. The lateral movement is typically a crucial stage of an APT attack. The system leveraged an attack specification that captures fundamental characteristics of lateral movement as a set of key path properties. which has been integrated into the anomaly detection algorithm. This rule-based anomaly scoring algorithm can surface the login paths that most likely reflect lateral movement via reconstructed global movement activity from point-wise login events.

The approach in [17] focuses on APT detection and addresses alert fatigue from a network direction perspective, referring to the movement of attack activity between network zones. They proposed APT scenario graphs derived from alerts. The algorithm for generating these graphs involves sequential steps: defining network directions inside zones according to Network Kill Chain State Machine (NKCSM) stages, assigning an APT stage to each alert and meta-alert, generating an alert graph based on timelines and recent infections, deriving an APT infection graph by aggregating and eliminating obsolete information to reduce graph density, and finally creating the APT scenario graph by extracting valid paths between stages.

Furthermore, attack specification designs have been integrated into other systems. Xiong et al. [103] proposed the concept of atomic suspicious indicators (ASIs) and incorporated them with rules and malicious states as configuration files in their built detection system. This integration achieved near zero false positives (FPs), as explained in their research.

Rule-based solutions have demonstrated exceptional performance in detecting majority of threats. However, they have some notable limitations. These include their reliance on post-event detection, as rules are typically created only after an attack has occurred. Additionally, they require significant manual effort to configure rules and gather related information. Perhaps most critically, they lack the ability to identify unknown attack patterns. [123] In contrast, machine learning application address these shortcomings effectively.

b) Machine Learning Applications

The category of detection using ML involves the implementation of ML algorithms and deep learning models to detect APTs. Those algorithms and models are widely used to learn feature spaces [124], [125] and find the correlation between different data sources. They primarily detect APTs-based anomaly detection, combining other auxiliary techniques to improve precision and reduce FPs. Numerous works have explored the application of ML in detecting APT activities.

Shen et al. [126] also utilized ML techniques to enhance APT detection in Internet of Things (IoT) environments. They proposed a Prior Knowledge Input (PKI) model, wherein an unsupervised clustering method was applied for pre-classifying the original dataset. The extracted prior knowledge was incorporated into the supervised models to reduce training complexity and determine the optimal mapping between the raw data and true labels. The study highlights the limitations of conventional machine learning strategies due to the scarcity of APT attack data and suggests that the PKI model can address these shortcomings by embedding prior knowledge into the detection process. However, potential limitations include the

reliance on the quality of clustering for prior knowledge generation, which may not always accurately reflect the true data distribution, and the challenge of handling the imbalance in datasets where normal traffic dominates, potentially affecting the model's ability to detect rare APT events.

To address enterprise-level lateral movement detection, King et al. [104] proposed a scalable temporal link prediction system called Euler. This approach integrates GNNs and recurrent neural networks (RNNs) using an encoder-decoder structure in a distributed manner. GNNs map network snapshots to low-dimensional embedding vectors, while RNNs encode the dynamic changes between connections. A leader machine controls the workflow and instructs multiple workers responsible for the parallel processing of network snapshots. This approach is effective in distinguishing dynamic network changes in different timestamps, offering an advantage over solutions focusing on static network status.

To preserve privacy during detection, Naseri et al. [127] investigated using FL to predict future security events. They developed CERBERUS, a system utilizing Recurrent Neural Networks (RNNs) from multiple organizations. The system underwent utility, robustness, and privacy testing through comparison experiments on non-independent-and-identically-distributed (non-IID) datasets, as well as adversary attacks and defense on models. Central differential privacy (CPD) in FL was applied to defend against inference attacks, achieved by sharing only dataset information. However, the paper also highlights challenges in deploying FL for predictive security, such as ensuring model robustness against adversarial attacks and maintaining high utility while preserving privacy.

Du et al. [105] presented a novel approach, called Vul-RAG, that utilized a knowledge-level retrieval-augmented generation (RAG) framework to enhance vulnerability detection in code. This approach operated in three phrases: constructing a vulnerability knowledge base from existing CVE instances, retrieving relevant vulnerability knowledge based on functional semantics for a given code snippet, and leveraging LLMs to assess the vulnerability of the code by reasoning about the causes and potential fixes. However, limitations include the potential failure of LLMs when handling lengthy contexts, as directly incorporating all retrieved knowledge items can hinder performance. Additionally, the approach relies heavily on the quality of the constructed knowledge base, which may not cover all possible vulnerabilities or scenarios.

Pasquale et al. [106] introduced ChainReactor, an automated system for discovering privilege escalation chains on Unix systems using AI planning. It addressed the limitation of current vulnerability research, which focuses on individual bugs rather than multi-step exploits. This system achieved the detection via extracting system capabilities and encoding them into a Planning Domain Definition Language (PDDL) problem, enabling the detection of both known and novel exploitation chains. However, it has several limitations: it currently focuses only on privilege escalation, supports a limited set of actions, and does not produce executable exploit code for validation.

Despite the effectiveness of deep learning solutions, they suffer from interpretability issues, being similar to black-box models [20], [128]. Bias may be introduced due to data quality and uncontrolled factors [10], leading to FPs in deep learning-based anomaly detection systems. To address potential bias and FPs, other solutions using pre-defined rules and scoring-based anomalies have been proposed in various works.

c) Scoring Algorithm

The scoring algorithm is a component adopted in various solutions [10], [90], [101]. It typically involves two types of scores: similarity scores and anomaly scores. Anomaly scores are used to aggregate anomaly or risk scores in most cases. On the other hand, similarity scores can be computed at the code-level or path-level.

As explored in Wang et al. [129], they tackled the tracking problem among devices and calculated similarity scores between devices for tracking purposes. Similarly, Yi et al. [90] compared code in forked projects with their original counterparts to identify the spread of vulnerabilities using similarity score computation.

In contrast, anomaly scores find broader application. Ho et al. [101] utilized a rule-based anomaly scoring algorithm to identify anomaly paths when locating APTs during the lateral movement stage. Another study by Hassan et al. [10] introduced a threat scoring methodology that enhances accuracy and performance in APT detection. Bhattarai et al. [8] defined risk scores based on different events in various layers of the network, calculated using rarity scores and sparsity scores of anomaly events. Additionally, intelligence-driven detection algorithms, proposed by Li et al. [20], employed risk scoring to identify anomalies based on risk thresholds.

Scoring algorithm-based methods help reduce false positives in machine learning applications and enhance explainability to some extent. However, as they operate as an additional processing layer on top of other solutions, they can increase computational overhead and introduce latency in decision-making.

3) Industrial Detection Solutions

As a supplement, we also provide industrial solution on how to detect and find the exploitation details of those APTs based on supply chain vulnerabilities in Table IV.

To summarize these solutions, we can divide them into following categories:

- Behavioral and Process-Based Detection: Solarwinds, NotPetya, Cozy Bear, and OilRig.
- Hash-Based and Artifact Detection: Kaseya VSA.
- Dynamic Code Execution Analysis: Lazarus Group.
- Abnormal Behavior and Traffic Analysis: CCleaner, and Codecov.

B. Censoring Methods

This category encompasses defense methods specifically targeting supply chain attacks, as presented in Table III based on Yang's Poirot [102]. The focus of these works is primarily on early detection at the point of infection, with an emphasis on code-level analysis. We have further divided this category into three subgroups to provide a more detailed classification of these defense methods.

TABLE IV: Collection of Industrial Detection Methods against Supply Chain based APT

Name	Detection Opportunities
SolarWinds [88]	config in SSL certificates, IP address and logon activity, certain patterns of SMB sessions
CCleaner [130]	DGA detection, abnormal behaviour, execution of shellcode
NotPetya [81]	special process execution, network propagation with SMB, unauthorized MBR modification
Kaseya VSA [82]	certain hashes of files and domains
Cozy Bear [131]	system binary proxy execution, logon auto-start execution
OilRig [132]	scheduled tasks, DNS protocol
Lazarus Group [133]	DLL side-loading, reflective DLL injection, DLL search order hijacking
Codecov [84]	dependencies, atypical IP addresses or regions, abnormal usage of env variables

1) Differential Analysis

Differential analysis systematically compares elements to identify patterns or changes. In software security, it involves comparing benign and infected software versions to detect malicious components, highlighting specific divergences and potential threats. This process helps identify anomalies indicating malicious code, enhancing software security and integrity. The systematic examination of variations between different software versions is instrumental in enhancing the overall security and integrity of SSC.

Barr-Smith et al. [5] deployed an approach to detect the insertion of malicious functionality in close-source SSCs through differential analysis of binaries. The differential analysis points at the comparison of binary analysis between build versions instead of analyzing a single version. By building the system Exorcist to automatically analyze two examples, the malicious indicators can be found via static analysis, dynamic analysis, and binary analysis. The outcome of analysed results are weighted in order to increase the reliability.

Froh et al. [109] used differential static analysis to identify malicious updates in open-source packages. They used CodeQL [5] to detect suspicious behavior by comparing the results between different versions of the packages. The features analyzed include network activity, process actions, obfuscation techniques, metadata, and code generation. Each update is assigned a score based on these features and, if the score exceeds a certain threshold, the update is flagged as malicious. However, a limitation of this approach could be its dependency on the accuracy and comprehensiveness of the CodeQL specifications, which may not cover all possible malicious behaviors or adapt quickly to new types of attacks.

Zheng et al. [134] designed a dataset named D2A to enhance AI-based vulnerability detection, using differential analysis to label issues identified by static analysis tools. D2A was constructed by analyzing version pairs from open-source projects, focusing on bug-fixing commits to identify real bugs that disappear post-commit, thus providing a more realistic dataset for training models in vulnerability identification. However, limitations include the continued reliance on static analysis tools, which may still produce false positives, and the dataset's focus on open-source projects, which might not fully generalize to proprietary software environments.

Differential analysis provides insights into how the current code or package differs from a benign version, enabling the identification of abnormal patterns to some extent. However, its effectiveness is limited by the requirement for a large number of benign examples. Additionally, it may produce false positives, as mismatches do not always equate to anomalies.

2) Program Analysis

Program analysis involves examining code using static and dynamic techniques. Static analysis reviews the code without running it to find potential issues and security risks. Dynamic analysis monitors the program during execution to observe behavior, such as function calls and memory usage, revealing runtime vulnerabilities that static analysis might miss. Together, these methods provide a thorough understanding of the program's security and potential risks.

Ladisa et al. [13] focuses on the supply chain of the Java ecosystem, driven by the scarcity of SSC studies in this specific domain. The authors proposed the static analysis approaches for malicious injected OSS Java repositories. The static solution focuses on the features extracted from constant pool and bytecode instructions with intra-procedural data-flow analysis. By extracting critical information, their solution achieved a promising detection rate with a less amount of data to review. The detection rate is higher than the low detection within antivirus (AV) methods. However, the paper acknowledges several limitations: the undecidability of determining maliciousness due to its relation to the Halting problem, the inherent constraints of static analysis, and the focus on specific types of malware, excluding others like hidden credentials or security check removals.

Yi et al. [90] introduced BlockScope to identify propagated vulnerabilities in forked blockchain programs. This tool analyzes patched code from original projects by extracting key statements using the Unix `grep` facility. By examining the surrounding lines of code, the boundaries of ks are identified. The vulnerability of a new code is determined by comparing its similarity score, calculated using a normalized Levenshtein algorithm, with a predefined threshold. BlockScope also efficiently narrows down the search scope by automatically extracting and using the context of the patch code.

Wang et al. [135] presented a novel approach to detecting security vulnerabilities in software by introducing a representation called a vulnerability net. This net integrated with data dependence graphs and control flow graphs to enhance the detection of taint-style vulnerabilities. The approach aimed to combine the strengths of static analysis tools and manual audits by incorporating auditor expertise into the analysis process. A limitation of this approach is that it may still require significant manual input to accurately model complex software behaviors, which can be time-consuming and may not fully eliminate the potential for human error.

[5]https://codeql.github.com/

Censoring-based methods primarily focus on analyzing and detecting threats by examining available source code. By extracting insights from malicious code, these methods significantly enhance the likelihood of detecting similar malicious code. Once deployed, incoming code can be quickly analyzed, and malicious code effectively identified. However, a critical challenge arises when source code is unavailable or inaccessible. This limitation has driven the development of proactive methods capable of detecting a broader range of threats from alternative perspectives beyond source code analysis.

3) Machine Learning

Machine learning can be applied at the feature level of code to detect specific patterns indicative of malicious behavior. By analyzing these patterns, ML algorithms can identify a range of threats, including Trojans [136], backdoors [107], and other malicious scripts [108].

Cheng et al. [108] proposed a forensic technique named BEAGLE to defend deep learning backdoor. This technique contains automatic attack root cause analysis, attack summarization, and scanner synthesis functions. These functions can decompose a Trojaned input into its original benign version and its malicious trigger. The decomposed information can help summarize the attack and also feed the scanner database.

Phillip et al. [107] explored backdoor mitigation in FL setting ups via deep model inspection. They proposed a novel model DeepSight which works as a filter composed of three techniques called cosine distances, NormalizED UPdate energies (NEUPs), and Division Differences (DDifs). These techniques perform characterizing the data distribution, and measuring fine-grained differences in the model internal structure and its outputs. Therefore, it is possible to identify suspicious model updates.

To address threats in OSS, Vasilakis et al. [15] proposed Active Library Learning and Regeneration (ALR) for inferring and regenerating the behavior of software components. Integrated into the Harp system, ALR improves string processing libraries by exploring, inferring, and regenerating their behavior. It models behavior using a domain-specific language (DSL) and creates vulnerability-free code that mimics the original. Harp also uses static and dynamic analysis to better understand the behavior of string processing components.

To detect vulnerabilities in function-level source code, Liu et al. [110] introduced Vul-LMGNNs, a novel model that combines sequence and graph embedding techniques. This model accepted the input of CPG representation of source code and leveraged a pre-trained Program Language (PL) model to extract local semantic features. Additionally, the model employed a Gated GNN equipped with convolutional layers to fuse heterogeneous information within this graph. However, the paper acknowledges two main limitations: the inherent limitation of GNNs in propagating structural knowledge across multiple layers, which affects the ability to capture long-range dependencies, and the under-utilization of the synergy between sequence-based and GNN approaches in current research.

The censoring based methods focus on the detection on available source code. By extracting insights from malicious code, the detection possibility for similar malicious code can become extremely high. When these methods are deployed, the new coming code can be quickly checked, and the malicious code can also be detected. However, one critical question is how these methods can work when the source code is not available or not accessible, which motivated the development of some proactive methods that can detect wide range of threats from other perspectives rather than source code.

C. Proactive Methods

The proactive method involves the deployment of defense technologies prior to the occurrence and/or detection of APT attacks. This approach can be approached from distinct angles. Firstly, it begins with assessing an institution's assets and identifying potential attack vectors targeting those assets. This includes collecting and analyzing public threat data and implementing typical defense solutions. These actions fall under the umbrella of risk management, encompassing impact analysis for owned code or projects, enumeration of attack vectors, and the utilization of cyber intelligence.

Another perspective of the proactive method focuses on prevention. One of the most efficient strategies found is the application of blockchain technology to ensure the verification of official digital sources or provenance data. By leveraging blockchain's immutable and transparent nature, organizations can enhance the trustworthiness and integrity of their digital assets, thereby mitigating the risks associated with potential APT attacks.

1) Risk Management

a) Impact Analysis

Impact analysis refers to the systematic evaluation of potential or realized consequences that threats pose to an organization's assets, operations, or overall security posture. This process involves identifying the vulnerabilities exploited by attackers, assessing the direct and indirect effects of such exploitation, and quantifying the resulting risks to prioritize remediation efforts.

In [16], the number of downloads has been mentioned as one of the attributes to measure the impact of influence of an OSS attack. Similarly, Jiang et al. [44] investigated the security of the Pre-trained Model (PTM) supply chain, focusing on models hosted on platforms like Hugging Face, where popular models see millions of downloads. Combining techniques from package ecosystem security and deep learning attacks, they analyzed threats and evaluated security risks across eight major model hubs. Their findings highlighted insufficient defenses for PTMs and proposed future directions, including dependency and impact analysis, along with automated tools for model auditing and scanning.

Yang et al. [137] proposed a risk management approach to mitigate APT-related losses. They modeled expected loss, state limitations, and response resources using a graphical topology of interactive hosts. Treating the problem as a game-theoretic challenge, they sought a Nash equilibrium, solved via a greedy algorithm to generate response strategies. The approach was tested with various attack/response strategies, evaluating potential losses in different scenarios, considering factors such as loss, value vectors, topology, and organizational context.

Guo et al. [138] explored how emerging information technologies (ITs) such as big data analytics, the Internet of Things, and blockchain can mitigate supply chain vulnerability in volatile environments. By identifying nine emerging ITs and fourteen mitigating factors, the research employed a novel DEMATEL-ISM approach combined with grounded theory to reveal multi-stage impact pathways. The findings suggest that these technologies do not directly reduce supply chain vulnerability but enhance internal factors by addressing external organizational factors, thereby strengthening supply chain resilience. However, the study's limitations include potential biases from the literature and interviews used to identify ITs and mitigating factors, and the generalizability of the findings may be constrained by the specific contexts.

Impact analysis provides a reliable prediction of potential impacts based on various factors, making it valuable for deploying prioritized defense strategies. However, these solutions require further evaluation in real-world scenarios to ensure the proposed models perform as intended.

b) Attack Vector Enumeration

Attack vector enumeration denotes the systematic identification and categorization of all possible pathways or methods that an adversary could exploit to compromise a system, application, or network within the supply chain. This process is critical for understanding and mitigating risks, as supply chains often involve multiple interconnected entities and systems, creating a wide range of potential entry points for attackers.

Ladisa et al. [16] introduced a comprehensive taxonomy of attacks, structured as an attack tree, which is independent of specific programming languages or ecosystems and encompasses all stages of the supply chain from code contributions to package distribution. The taxonomy's validity and utility were positively evaluated through surveys with 17 domain experts and 134 software developers, who also assessed the practicality and costs of the safeguards.

To measure the impact of supply chain attacks on package managers, Duan et al. [14] proposed a framework to assess the functional and security features of package managers, analyzing those for interpreted languages using program analysis techniques. They identified malicious programs in PyPI, NPM, and RubyGems but faced limitations, including low coverage in dynamic analysis, inaccuracies in static analysis, and a narrow focus on interpreted languages without offering comprehensive solutions to supply chain attacks.

Eggers et al. [22] investigated the cyber-attack surface related to nuclear-related supply chains. The authors discussed overlooked supply chain threats and vulnerabilities and designed an attack surface diagram to enumerate risks and potential attacks between different steps. The diagram aids security personnel in enhancing risk management processes. This research serves as a platform for further supply chain research and can help develop supply chains with provenance capabilities. It also motivates the development of improved security credential verification processes and secure tamper-proof distribution methods.

Due to the complex inter-dependencies among numerous supply chain stakeholders, challenges arise, including the lack of third-party audit schemes and cascading cyber threats. Yeboah et al. [26] analyzed supply chain cyber threats using the STIX threat model in a case study of an Electric Power Station. By applying CVSS concepts, they quantified factors like Penetration and Manipulation to calculate attack probabilities and assess severity.

Enumerating attack vectors requires both expert knowledge and significant manual effort. By identifying these vectors, organizations can deploy more proactive and tailored defense methods, effectively preventing the majority of potential attack scenarios. However, a key drawback is the reliance on enumerated threats, which leaves unknown threats unaddressed.

c) Cyber Intelligence

Cyber intelligence in general refers to the collection, analysis, and interpretation of information related to specific cyber threats. To defend APTs based on SCVs, it involves using actionable insights to identify, monitor, and mitigate risks associated with both known and emerging vulnerabilities in the interconnected systems, vendors, and software within the supply chain.

Li et al. [20] proposed a cyber intelligence-driven approach for detecting APTs at the network edge. They developed a Bayesian Stackelberg game model and a CTI-based defense strategy, along with algorithms for dynamic resource allocation and real-time detection using deep reinforcement learning. Their work enhanced detection accuracy and transparency while addressing limitations in existing methods.

Ren et al. [113] designed a cybersecurity platform to enhance the attribution of APT using a knowledge graph model. This model is built from open source cyber threat intelligence (OSCTI) and aims to improve the way threat knowledge is expressed, allowing security researchers to efficiently access diverse threat information for intelligent decision making. The platform integrated deep learning and expert knowledge to continuously update and complete the knowledge graph, enabling proactive defense strategies rather than traditional passive methods.

Yin et al. [114] proposed a novel approach using graph-driven intelligence, arming the discovery of co-exploitation behavior as a link prediction problem within a vulnerability knowledge graph, to addresses the challenge of co-exploitation behavior in cybersecurity. They introduced a modality-aware graph convolutional network (MAGCN) to integrate multi-modal entity attributes and graph connectivity features into a unified feature space, enhancing link prediction performance. However, a limitation of this study is the reliance on historical data, which may not fully capture emerging vulnerabilities or novel exploitation techniques, potentially affecting the model's applicability to future threats.

A key challenge in integrating cyber intelligence into detection is that OSCTI may not always deliver comprehensive or up-to-date information, as highlighted in [113], [114]. To address this limitation, the incorporation of additional techniques is essential to ensure robust and proactive methods.

2) Block-Chain based Design

The integration of blockchain technology into the supply chain has been widely explored. In the survey [139], the authors explored solutions to solve issues within Supply Chain Management systems. Those issues relate to security, fraud,

traceability, and product counterfeiting. Utilizing blockchain can minimize the risks existing in supply chain. The following work provide more specific scenarios when integrating blockchain to solve potential threats.

Bandara et al. [140] proposed Let'sTrace, a system that combines blockchain, the Update Framework (TUF), In-Toto, and FL to increase cybersecurity in the CSC. TUF secures update systems, while In-Toto ensures a safe software development lifecycle. Blockchain, integrated hierarchically among stakeholders with smart contracts, verifies identities and messages. TUF and In-Toto further ensure update and supply chain integrity. Using FL, the system privately analyzes update and supply chain data, significantly reducing risks from compromised signing keys, repositories, third-party software, and update mechanisms.

Craciun et al. [30] proposed another possible solution to prevent the stealing of signatures in hardware development is the deployment of a blockchain infrastructure with smart contracts. This approach verifies trusted counterparts during integration by using encryption methods with public and private keys shared between senders and receivers.

Islam et al. [112] combined Physical Unclonable Function (PUF) enabled Radio Frequency Identification (RFID) tag with the support of blockchain to prevent product counterfeiting issue. They proposed mainly three protocols: registration protocol, verification protocol, and transaction protocol to ensure the authentication of ownership and also the safety of sensitive information. Similarly, in [111], the authors also utilized blockchain, especially emphasizing its decentralization, to provide the data provenance tracking and further ensure the data integrity. They have claimed this implementation can prevent single point failure and protect the whole supply chain from threats like Man-in-the-middle relay attack, denial-of-service, and spoofing attack on data of product records.

Blockchain-based methods take a preventive approach to safeguard the entire supply chain from tampering. However, their implementation faces significant challenges, including legal concerns and substantial engineering efforts, which have limited their practical adoption.

VI. Quantitative Analysis of Solutions

To validate the solutions against APTs based on SCVs, we perform quantitative analysis for them. To quantify the risk that an APT will succeed in progressing to the next stage without detection, we propose survival analysis [141], especially Cox Proportional Hazards models [142], based method to validate the efficiency of solutions against APTs , which is motivated by the work [143]. This analysis method was originally applied in medical domain to predict the survival rate of person at certain time, in which extra factors are considered. To combat APT activities, we incorporate this concept into measuring the duration of attack activities. This approach evaluates the potential for threat mitigation or detection at each stage of the attack by integrating multiple factors.

The factors to consider in this quantitative analysis are categorized into three groups: focused detection stage, system performance (like generality, false positive rate, etc), and applied number of core technologies. The first factor regarding focused stages is based on the assumption that earlier the detection and mitigation can take place, the less damage ensues. Each stage in APT depends on the previous stage, except that some stages have bidirectional dependencies due to iteration. It is also more feasible to identify APT in early stages than in later stages. Furthermore, when more stages are covered in analysis, there is a higher probability of precisely locating and mitigating APTs [144].

The detection performance is another factor that can potentially influence the outcome. One typical reason is the false positives, leading to alert fatigue. Furthermore, whether the detection system can adapt to newly or unknown exploitation methods and consistently evolve is also crucial in practice because that the attack technology has been evolving over years. Referring to Table VIII, we compare some recent work on supply chain security on metrics such as generality and scalability, from which we can identify the diverse capability of different solutions.

The number of technologies applied can indicate whether the solution combines multiple technologies, which can enhance detection performance and potentially overcome the drawbacks of the sole solution. However, the choice is also a trade-off between efficiency (compute overhead) and detection performance (accuracy). From Table III, we can reasonably infer the number of implemented technology falls inside the range from 1 to 4.

To calculate the detection "survival" at different APT stages, we construct a survival model that incorporates these dimensions as co-variates. Let S_i represent the APT stage, where $i \in \{1, 6\}$. Generality G reflects how well the detection solution can adapt and update over time, which is represented as a continuous score $G \in [0, 1]$. We take the same representation method for false positive rate (FP), for which $FP \in [0, 1]$. The number of core technology T is a discrete variable which falls into the range $T \in \{1, 4\}$. Finally, the survival function $S(t)$ represents the probability that a detection solution successfully detects an APT before stage t. A lower survival probability implies that APT has not been detected before a particular stage. The hazard function $h(t)$, motivated by [145], defines the instantaneous risk of failure at time t, expressing as:

$$h(t) = \lambda(t \mid G, FP, T) \cdot f\left(S_c\right) \tag{1}$$

The $\lambda_0(t)$ is the baseline hazard at time t and $f\left(S_c\right)$ is a stage coverage factor where S_c is the set of stages being monitored. The two separate functions are detailed as:

$$\lambda(t \mid G, FP, T) = \lambda_0(t) \exp\left(\beta_1 G + \beta_2 FP + \beta_3 T\right) \tag{2}$$

in which all the βs represent the coefficients that measure the impact of each factor on the hazard. We initialize the values for them like -0.8 (high generality lowers the hazard), 1.5 (higher false positive rate increases the hazard), and -0.3(more technologies slightly lower the hazard).

$$f(S_c) = 1 + \alpha \cdot \frac{|S_{\mathrm{c}}|}{S_{\mathrm{total}}} \tag{3}$$

TABLE V: Proposed Challenges in References

Proposed Challenges	limited stages detection	computation overhead	false positives	restricted precision	in-time update	scalability	entropy analysis	uncertainty	finite running	long-time attack	unknown attacks	supply chain attacks	evasion	sole data source	sole group of features	comparative features	sole system	novel exploitation
Du *et al.* (2024) [105]	✓	×	✓	✓	×	×	×	×	×	×	×	×	×	×	×	×	×	×
Pasquale *et al.* (2024) [106]	✓	×	×	✓	×	×	×	×	×	×	×	×	×	×	×	×	×	×
Liu *et al.* (2024) [110]	✓	×	×	×	×	×	×	×	×	✓	×	×	×	×	×	×	×	×
Bhattarai *et al.* (2022) [8]	×	×	×	×	×	×	×	×	×	✓	✓	✓	✓	×	×	×	×	×
Irshad *et al.* (2021) [9]	✓	✓	×	×	×	×	×	×	×	×	×	✓	✓	×	×	×	×	×
Alageel *et al.* (2021) [6]	×	×	✓	×	×	×	×	×	×	×	×	✓	✓	×	×	✓	×	×
Alsaheel *et al.* (2021) [7]	×	×	×	✓	×	×	×	×	×	×	×	✓	✓	✓	×	×	×	×
Ho *et al.* (2021) [101]	×	×	×	×	×	×	×	×	×	×	✓	×	×	×	×	×	×	×
Wilkens *et al.* (2021) [17]	×	×	✓	×	×	×	×	×	×	×	×	✓	✓	✓	×	×	×	×
King *et al.* (2023) [104]	✓	×	×	×	×	×	×	×	×	×	×	✓	✓	×	×	×	×	×
Xiong *et al.* (2022) [103]	×	×	×	×	×	×	×	×	×	×	✓	✓	✓	×	×	×	×	×
Barr-Smith *et al.* (2022) [5]	×	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	×	✓	×
Ladisa *et al.* (2022) [13]	×	×	×	×	×	×	✓	✓	×	×	×	×	✓	×	×	×	×	✓
Vasilakis *et al.* (2021) [15]	×	✓	×	×	×	×	×	×	✓	×	×	×	✓	×	✓	×	×	×

The $|S_c|$ is the number of stages the solution monitors. The α is a weighting factor indicating the impact of each additional stage on the detection likelihood, with default value 0.5.

The solutions chosen to calculate the quantitative hazard score are from Tables VIII and III with their drawbacks and the challenges described as examples to represent the detection performance score. In term of the initial baseline hazard for various stages, we propose a stage-based difficulty weight $w(t)$,inspired by Growth Models [146]. This weight is assigned to each stage under the assumption that earlier stages have lower values due to the absence of overt malicious activity [147]. The weight is computed using the following formula:

$$w(i) = \alpha \cdot i^b \Rightarrow \lambda(i) = \lambda_0 \cdot w(i) \tag{4}$$

Let i as the index of stage (2 for delivery, etc), and a (0.2) and b (1.5) are constants that control the steepness and shape of the increase across stages. The Table VI illustrates the quantitative score after taking described factors into consideration, to validate the detection performance against APTs. To quantify factors like generality and false positives, we map the string values defined based on evidence from papers to numeric values to build clear criteria. The mapping values for generality are like:

- High (0.8 - 1.0): The solution is demonstrated to work across multiple domains or threat models, supported by diverse experimental results, evaluations, or claims.
- Medium (0.4-0.7): The solution is effective in multiple but related domains or for a limited range of threats with moderate generalization.
- Low (0-0.3): The solution focuses on highly specific scenario or dataset with limited applicability to other domains or threats.

The mapping values for scalability are like:

- Low (0-0.3): False positive rate $\leq 10\%$, based on empirical evaluation or clear metrics or claimed.
- Medium (0.4-0.7): False positive rate between 10-30%.
- High (0.8-1.0): False positive rate $\succ 30\%$.

The goal is to provide a qualitative comparison informed by quantitative analysis, rather than aiming for precise estimates.

TABLE VI: Quantitative Analysis of Solutions Against APTs

Solutions	Focused Stages	Generality	False Positives	Core Methods	Hazard Score
Du *et al.* (2024) [105]	delivery	0.9	0.3	1	0.346
Pasq. *et al.* (2024) [106]	lateral movement	0.3	0.1	4	0.102
King *et al.* (2023) [104]	lateral movement	0.6	0.2	2	0.169
Ladisa *et al.* (2023) [71]	delivery	0.6	0.2	1	0.254
Xiong *et al.* (2022) [103]	full stages	0.8	0	2	0.518
Barr. *et al.* (2022) [5]	delivery	0.7	0.2	1	0.254
Alsah. *et al.* (2021) [7]	full stages	0.7	0.2	2	0.518

The table above provides an estimate of the likelihood of detection failure for each solution, with higher values indicating a greater chance of failure. In recent years, the hazard score has decreased to below 0.4, suggesting overall improvement in detection performance over time. However, a review of the literature reveals persistent critical challenges, highlighting potential opportunities for further research.

VII. Challenges & Research Opportunities

The latest research regarding specific sub-direction of exploiting SCVs has been discussed in this survey. As described in Table III, we find that recent work has made strides in addressing certain potential challenges. In particular, although some works may not explicitly mention full-stage APT detection, they demonstrate the capability to detect and track entire stages of APT attack [7]–[10], [17]. However, it is essential to acknowledge that, achieving full-stage detection can introduce biases and side effects on performance in practice.

To provide an updated overview of current research challenges, we have documented the limitations and key issues, as illustrated in Table V and Table VIII. Enriching the context, we have considered both the supply chain and evasion attack perspectives. The supply chain aspect contributes to addressing typical types of APT attacks, while evasion attacks focus on potential adversarial attacks for data falsification, which can undermine defense systems. In the following discussion, we highlight key challenges identified from the reviewed papers:

Supply Chain-based APT Datasets: Current detection systems are primarily focused on simulated APT attacks that replicate common threats over a short time frame [8], [148]. However, real-world APT activities can extend for several months [8], [88], and SCV-based APTs are particularly difficult to detect [4], [78]. Additionally, latest dataset is not representative to demonstrate the exploration on advanced supply chains [68], [72], which further causes the lack of benchmark comparison on detection performance for this specific threats [71]. Those limitations from available data sets can hinder the detection performance for these specific types of APT.

The further research direction should involve the generation of new dataset can illustrate the latest trend for supply chain exploration, especially the scenarios since Solarwinds happened [4], which can be served for benchmark of works under this attack.

Multi-Source Fusion: Previous works in the field have predominantly relied on a single data source, such as browser-based history [119], audit logs [9], or network traffic [6], as the cornerstone of their data provenance-based approaches [7], [17]. Although some efforts, such as [7], [44], have integrated two data sources, such as system and application logs, these sources still fall into the same category of low-level logs, potentially lacking insights into higher-level behaviors. The solutions involve fusing multi-source data, such as system, network, process information, to comprehensive represent the system status, potentially using a graph-based approach. Furthermore, incorporating temporal information into the graph representation is essential to capture the evolving status of system components. This enables the use of temporal graph learning [149] to identify evolving anomalies effectively.

Evasion and Privacy: While numerous efficient systems demonstrate remarkable detection performance, some still lack robust defenses against potential evasion attacks. Threats such as adversarial attacks for data, model stealing, and privacy disclosure can pose serious challenges [8], [101], [129]. Certain initiatives have focused on addressing privacy and robustness issues [127], [140]. However, these solutions remain challenging to deploy in real-world applications when it comes to legal issues or trust issues, even with the possibility of reduced performance. The integration of FL presents a viable solution to address potential privacy issues [68], [150], especially when it is not feasible to transmit client monitoring data to a central server for analysis. To counter evasion threats, augmenting training data with adversarial examples or noise is a recommended approach for enhancing robustness and mitigating bias during the training process [151], [152].

Self-Evolving Systems: Most current research on APTs and supply chain threats emphasizes achieving high precision and other metrics on specific datasets, with some approaches demonstrating generalizability to new patterns and potential unknown threats [76], [129]. However, these solutions often overlook the need for systems that evolve over time. A critical gap remains in how to incorporate information from false positives and negatives to continuously update and improve threat detection systems. While continual learning has been studied for years, with methods proposed to mitigate catastrophic forgetting [153], [154], this approach deserves greater attention in the design of threat detection frameworks.

Efficient Explainable Detection: With advancements in LLM, recent studies [67], [110] have begun integrating LLMs into threat detection, offering potentially more explainable results. However, due to their high computational overhead and longer training time, deploying LLMs for APT detection remains impractical. Graph-based representations for analyzing software or function dependencies also enhance explainability, yet integrating of GNN [66], [73] further escalates computational demands. Future research should concentrate on development of lightweight LLM [155], efficient training or retraining for LLM, and applying model compression techniques to GNNs [156] or exploring distributed training for GNNs.

VIII. Conclusion

In this survey, we focus on a special type of APT that exploits SCVs. This is a subject that, to the best of our knowledge, has not been thoroughly researched before. We gather real-world APT attack scenarios and analyze both industrial detection and academic defense methods to provide a comprehensive overview of the current challenges, solutions, and future directions in this domain. Employing various analysis techniques such as graph-based analysis, statistical analysis, and comparative studies, we extract valuable insights from the collected data. To better comprehend the current threat landscape, we present a triage of supply chain-based threats based on different frameworks. Additionally, we propose a unified threat model based on actual scenarios and develop a customized detection chain. By combining hardware and SSC perspectives, we conduct an in-depth enumeration of attack vectors, offering guidance for defense deployment. To showcase state-of-the-art defense methodologies, we further classify defense technologies pertaining to both APTs and supply chain attacks. Finally, we identify limitations and drawbacks from existing literature, elucidating the challenges and outlining future research directions. Our intention is for this survey to benefit both industry and academia and contribute to safeguarding the Internet world against APTs exploiting SCVs.

Acknowledgments

This work is partially funded by the EU Horizon grants TRACE #101104278 and COCOON #101120221, and Jumpsec Ltd. PhD Scholarship.

References

[1] Mandiant, “Special report: Mandiant m-trends 2024,” 2024. [Online]. Available: https://services.google.com/fh/files/misc/m-trends-2024.pdf

[2] OWASP.org, "Owasp top 10 for llm applications," 2023. [Online]. Available: https://owasp.org/www-project-top-10-for-large-language-model-applications/assets/PDF/OWASP-Top-10-for-LLMs-2023-v1_0_1.pdf

[3] European Union Agency for Cybersecurity., *ENISA Threat Landscape for Supply Chain Attacks.* LU: Publications Office, 2021.

[4] S. Ozarslan, "Tactics, techniques, and procedures (ttps) used in the solarwinds breach," 2020. [Online]. Available: https://www.picussecurity.com/resource/blog/ttps-used-in-the-solarwinds-breach

[5] F. Barr-Smith, T. Blazytko, R. Baker, and I. Martinovic, "Exorcist: Automated Differential Analysis to Detect Compromises in Closed-Source Software Supply Chains," in *Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*. Los Angeles CA USA: ACM, 2022, pp. 51–61.

[6] A. Alageel and S. Maffeis, "Hawk-Eye: Holistic detection of APT command and control domains," in *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. Virtual Event Republic of Korea: ACM, 2021, pp. 1664–1673.

[7] A. A. Alsaheel, Y. Nan, S. Ma, L. Yu, G. Walkup, Z. B. Celik, X. Zhang, and D. Xu, "Atlas: A sequence-based learning approach for attack investigation," in *USENIX Security Symposium*, 2021.

[8] B. Bhattarai and H. Huang, "SteinerLog: Prize Collecting the Audit Logs for Threat Hunting on Enterprise Network," in *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*. Nagasaki Japan: ACM, 2022, pp. 97–108.

[9] H. Irshad, G. Ciocarlie, A. Gehani, V. Yegneswaran, K. H. Lee, J. Patel, S. Jha, Y. Kwon, D. Xu, and X. Zhang, "TRACE: Enterprise-Wide Provenance Tracking for Real-Time APT Detection," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4363–4376, 2021.

[10] W. U. Hassan, A. Bates, and D. Marino, "Tactical Provenance Analysis for Endpoint Detection and Response Systems," in *2020 IEEE Symposium on Security and Privacy (SP)*. San Francisco, CA, USA: IEEE, 2020, pp. 1172–1189.

[11] T. Sobb, B. Turnbull, and N. Moustafa, "Supply Chain 4.0: A Survey of Cyber Security Challenges, Solutions and Future Directions," *Electronics*, vol. 9, no. 11, p. 1864, 2020.

[12] A. Yeboah-Ofori and C. Boachie, "Malware Attack Predictive Analytics in a Cyber Supply Chain Context Using Machine Learning," in *2019 International Conference on Cyber Security and Internet of Things (ICSIoT)*. Accra, Ghana: IEEE, 2019, pp. 66–73.

[13] P. Ladisa, H. Plate, M. Martinez, O. Barais, and S. E. Ponta, "Towards the Detection of Malicious Java Packages," in *Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*. Los Angeles CA USA: ACM, 2022, pp. 63–72.

[14] R. Duan, O. Alrawi, R. P. Kasturi, R. Elder, B. Saltaformaggio, and W. Lee, "Towards measuring supply chain attacks on package managers for interpreted languages," *Proceedings 2021 Network and Distributed System Security Symposium*, 2020.

[15] N. Vasilakis, A. Benetopoulos, S. Handa, A. Schoen, J. Shen, and M. C. Rinard, "Supply-Chain Vulnerability Elimination via Active Learning and Regeneration," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. Virtual Event Republic of Korea: ACM, 2021, pp. 1755–1770.

[16] P. Ladisa, H. Plate, M. Martinez, and O. Barais, "Taxonomy of Attacks on Open-Source Software Supply Chains," 2022.

[17] F. Wilkens, F. Ortmann, S. Haas, M. Vallentin, and M. Fischer, "Multi-Stage Attack Detection via Kill Chain State Machines," in *Proceedings of the 3rd Workshop on Cyber-Security Arms Race*. Virtual Event Republic of Korea: ACM, 2021, pp. 13–24.

[18] T. Steffens, *Attribution of Advanced Persistent Threats: How to Identify the Actors Behind Cyber-Espionage*, 01 2020.

[19] A. Alshamrani, S. Myneni, A. Chowdhary, and D. Huang, "A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1851–1877, 2019.

[20] H. Li, J. Wu, H. Xu, G. Li, and M. Guizani, "Explainable Intelligence-Driven Defense Mechanism against Advanced Persistent Threats: A Joint Edge Game and AI Approach," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2021.

[21] L. Caviglione, M. Podolski, W. Mazurczyk, and M. Ianigro, "2. covert channels in personal cloud storage services: The case of dropbox," *IEEE Transactions on Industrial Informatics*, 2017.

[22] S. Eggers, "A novel approach for analyzing the nuclear supply chain cyber-attack surface," *Nuclear Engineering and Technology*, vol. 53, no. 3, pp. 879–887, 2021.

[23] M. J. Hossain Faruk, M. Tasnim, H. Shahriar, M. Valero, A. Rahman, and F. Wu, "Investigating novel approaches to defend software supply chain attacks," in *2022 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, 2022, pp. 283–288.

[24] H. Siadati, S. Jafarikhah, E. Sahin, T. B. Hernandez, E. L. Tripp, D. Khryashchev, and A. Kharraz, "Devphish: Exploring social engineering in software supply chain attacks on developers," 2024.

[25] M. O. Ibiyemi and D. O. Olutimehin, "Cybersecurity in supply chains: Addressing emerging threats with strategic measures," *International journal of management & entrepreneurship research*, vol. 6, no. 6, pp. 2024–2047, 2024.

[26] A. Yeboah-Ofori and S. Islam, "Cyber Security Threat Modeling for Supply Chain Organizational Environments," *Future Internet*, vol. 11, no. 3, p. 63, 2019.

[27] I. Dmitry, "Managing Risks in Supply Chains with Digital Twins and Simulation." [Online]. Available: https://scms-summit.com/media/9417/managing-risks-in-supply-chains-with-digital-twins-and-simulation.pdf

[28] K. R. Ludvigsen, S. Nagaraja, and A. Daly, "Preventing or Mitigating Adversarial Supply Chain Attacks: A Legal Analysis," in *Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*. Los Angeles CA USA: ACM, 2022, pp. 25–34.

[29] N. Gupta, A. Tiwari, S. T. S. Bukkapatnam, and R. Karri, "Additive manufacturing cyber-physical system: Supply chain cybersecurity and risks," *IEEE Access*, vol. 8, pp. 47 322–47 333, 2020.

[30] V. Craciun, P. A. Felber, A. Mogage, E. Onica, and R. Pires, "Malware in the SGX supply chain: Be careful when signing enclaves!" *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2020.

[31] M. Sánchez-Gordón and R. Colomo-Palacios, "Security as culture: A systematic literature review of devsecops," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, ser. ICSEW'20. New York, NY, USA: Association for Computing Machinery, 2020, p. 266–269.

[32] A. Saputra, E. Suryani, and N. A. Rakhmawati, "The robustness of machine learning models using mlsecops: A case study on delivery service forecasting," in *2023 14th International Conference on Information & Communication Technology and System (ICTS)*, 2023, pp. 265–270.

[33] K. Grosse, L. Bieringer, T. R. Besold, B. Biggio, and K. Krombholz, "Machine learning security in industry: A quantitative survey," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 1749–1762, 2023.

[34] S. Mäkinen, H. Skogström, E. Laaksonen, and T. Mikkonen, "Who needs mlops: What data scientists seek to accomplish and how can mlops help?" 2021.

[35] I. Hepworth, K. Olive, K. Dasgupta, M. Le, M. Lodato, M. Maruseac, S. Meiklejohn, S. Chaudhuri, and T. Minkus, "Securing the ai software supply chain," Google, Tech. Rep., 2024.

[36] M. Jegorova, C. Kaul, C. Mayor, A. Q. O&x27;Neil, A. Weir, R. Murray-Smith, and S. A. Tsaftaris, "Survey: Leakage and privacy at inference time," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 45, no. 07, pp. 9090–9108, jul 2023.

[37] K.-F. Cheung, M. G. Bell, and J. Bhattacharjya, "Cybersecurity in logistics and supply chain management: An overview and future research directions," *Transportation Research Part E-logistics and Transportation Review*, vol. 146, pp. 102 217–, 2021.

[38] K. Stouffer, T. Zimmerman, C. Tang, J. Lubell, J. Cichonski, M. Pease, and J. McCarthy, "Cybersecurity Framework Version 1.1 Manufacturing Profile," National Institute of Standards and Technology, Tech. Rep., 2020.

[39] MITRE, "Adversarial tactics, techniques, & common knowledge (atlas)," 2024. [Online]. Available: https://atlas.mitre.org/matrices/ATLAS

[40] N. Moghadasi, M. N. Luu, R. O. Adekunle, T. L. Polmateer, M. C. Manasco, J. M. Emmert, and J. H. Lambert, "Research and development priorities for security of embedded hardware devices," *IEEE Transactions on Engineering Management*, pp. 1–12, 2022.

[41] RyotaK, "Tampering with arbitrary packages in @types scope of npm," 2021. [Online]. Available: https://blog.ryotak.net/post/definitelytyped-tamper-with-arbitrary-packages-en/

[42] D. L. Vu, I. Pashchenko, F. Massacci, H. Plate, and A. Sabetta, "Towards Using Source Code Repositories to Identify Software Supply Chain Attacks," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. Virtual Event USA: ACM, 2020, pp. 2093–2095.

[43] M. Ohm, H. Plate, A. Sykosch, and M. Meier, "Backstabber's Knife Collection: A Review of Open Source Software Supply Chain Attacks," 2020.

[44] W. Jiang, N. Synovic, R. Sethi, A. Indarapu, M. Hyatt, T. R. Schorlemmer, G. K. Thiruvathukal, and J. C. Davis, "An Empirical Study of Artifacts and Security Risks in the Pre-trained Model Supply Chain," in *Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*. Los Angeles CA USA: ACM, 2022, pp. 105–114.

[45] Y. Li and L. Xu, "Cybersecurity investments in a two-echelon supply chain with third-party risk propagation," *International Journal of Production Research*, vol. 59, no. 4, pp. 1216–1238, 2021.

[46] P. Banerjee, "Crowdstrike cyber incident vs. past major cyber incidents: Analysis and solutions," *International Journal For Multidisciplinary Research*, vol. 6, no. 4, 2024.

[47] M. N., N. D.R., B. E. Reddy, R. Buyya, V. K.R., S. S. Iyengar, and L. Patnaik, "Secure pharmaceutical supply chain using blockchain in iot cloud systems," *Internet of things*, 2024.

[48] P. Ladisa, S. E. Ponta, A. Sabetta, M. Martínez, and O. Barais, "Journey to the center of software supply chain attacks," *IEEE Security & Privacy*, vol. 21, no. 6, pp. 34–49, 2023.

[49] T. Liu, F. Lin, Z. Wang, C. Wang, Z. Ba, L. Lu, W. Xu, and K. Ren, "Magbackdoor: Beware of your loudspeaker as a backdoor for magnetic injection attacks," in *2023 IEEE Symposium on Security and Privacy (SP)*, 2023, pp. 3416–3431.

[50] D. Ibdah, N. Lachtar, A. A. Elkhail, A. Bacha, and H. Malik, "Dark firmware: A systematic approach to exploring application security risks in the presence of untrusted firmware," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*. San Sebastian: USENIX Association, Oct. 2020, pp. 413–426.

[51] Z. Pan, W. Shen, X. Wang, Y. Yang, R. Chang, Y. Liu, C. Liu, Y. Liu, and K. Ren, "Ambush from all sides: Understanding security threats in open-source software ci/cd pipelines," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–16, 2023.

[52] R. A. Lika, D. Murugiah, S. N. Brohi, and D. Ramasamy, "Notpetya: Cyber attack prevention through awareness via gamification," in *2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE)*, 2018, pp. 1–6.

[53] R. Alkhadra, J. Abuzaid, M. AlShammari, and N. Mohammad, "Solar winds hack: In-depth analysis and countermeasures," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2021, pp. 1–7.

[54] J. H. Lambert, J. M. Keisler, W. E. Wheeler, Z. A. Collier, and I. Linkov, "Multiscale approach to the security of hardware supply chains for energy systems," *Environment Systems and Decisions*, vol. 33, no. 3, pp. 326–334, 2013.

[55] L. Aniello, B. Halak, P. Chai, R. Dhall, M. Mihalea, and A. Wilczynski, "Anti-bluff: towards counterfeit mitigation in ic supply chains using blockchain and puf," *International Journal of Information Security*, vol. 20, no. 3, pp. 445–460, 2021.

[56] M. Ghaznavi, E. Jalalpour, M. A. Salahuddin, R. Boutaba, D. Migault, and S. Preda, "Content delivery network security: A survey," *IEEE Communications Surveys and Tutorials*, vol. 23, no. 4, pp. 2166–2190, 2021.

[57] Q. Su, Y. Shi, Y. Gao, T. S. Arthanari, and M. Wang, "The improvement of logistics management in china: A study of the risk perspective," *Sustainability*, 2024.

[58] K. E. Stecke and S. Kumar, "Sources of supply chain disruptions, factors that breed vulnerability, and mitigating strategies," *Journal of Marketing Channels*, vol. 16, no. 3, pp. 193–226, 2009.

[59] C. Lamb and S. Zacchiroli, "Reproducible builds: Increasing the integrity of software supply chains," *IEEE Software*, vol. 39, no. 2, pp. 62–70, 2022.

[60] M. D. Karumanchi, J. Sheeba, and S. Pradeep Devaneyan, "Integrated internet of things with cloud developed for data integrity problems on supply chain management," *Measurement: Sensors*, vol. 24, p. 100445, 2022.

[61] N. Patel, P. Krishnamurthy, S. Garg, and F. Khorrami, "Bait and switch: Online training data poisoning of autonomous driving systems," 2020.

[62] A. A. Pammu, K.-S. Chong, Y. Wang, and B.-H. Gwee, "A highly efficient side channel attack with profiling through relevance-learning on physical leakage information," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 3, pp. 376–387, 2019.

[63] M. Rigaki and S. Garcia, "A survey of privacy attacks in machine learning," *ACM Comput. Surv.*, vol. 56, no. 4, nov 2023. [Online]. Available: https://doi.org/10.1145/3624010

[64] Y. Siwakoti, M. Bhurtel, D. B. Rawat, and A. Oest, "Advances in iot security: Vulnerabilities, enabled criminal services, attacks, and countermeasures," *IEEE Internet of Things Journal*, vol. 10, pp. 11 224–11 239, 2023.

[65] M. Borhani, G. S. Gaba, J. Basaez, I. Avgouleas, and A. Gurtov, "6. a critical analysis of the industrial device scanners' potentials, risks, and preventives," *Journal of Industrial Information Integration*, 2024.

[66] H.-C. Tran, A.-D. Tran, and K.-H. Le, "DetectVul: A statement-level code vulnerability detection for Python," *Future Generation Computer Systems*, p. 107504, Sep. 2024.

[67] G. Lu, X. Ju, X. Chen, W. Pei, and Z. Cai, "Grace: Empowering llm-based software vulnerability detection with graph structure and in-context learning," *Journal of Systems and Software*, vol. 212, p. 112031, 2024.

[68] I. A. Khan, N. Moustafa, D. Pi, Y. Hussain, and N. A. Khan, "Dff-sc4n: A deep federated defence framework for protecting supply chain 4.0 networks," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 3, pp. 3300–3309, 2023.

[69] M. N., N. D.R., B. E. Reddy, R. Buyya, V. K.R., S. Iyengar, and L. Patnaik, "Secure pharmaceutical supply chain using blockchain in iot cloud systems," *Internet of Things*, vol. 26, p. 101215, 2024.

[70] R. Cerchione, P. Centobelli, and A. Angelino, "Blockchain-based iot model and experimental platform design in the defence supply chain," *IEEE Internet of Things Journal*, vol. 10, no. 24, pp. 22 033–22 039, 2023.

[71] P. Ladisa, S. E. Ponta, N. Ronzoni, M. Martinez, and O. Barais, "On the Feasibility of Cross-Language Detection of Malicious Packages in npm and PyPI," in *Annual Computer Security Applications Conference*, Dec. 2023, pp. 71–82.

[72] C. Liang, Q. Wei, Z. Jiang, Y. Wang, and J. Du, "A source code vulnerability detection method based on adaptive graph neural networks," in *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering Workshops*, ser. ASEW '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 187–196.

[73] M. F. Rozi, T. Ban, S. Ozawa, A. Yamada, T. Takahashi, and D. Inoue, "Securing Code with Context: Enhancing Vulnerability Detection through Contextualized Graph Representations," *IEEE Access*, pp. 1–1, 2024.

[74] H. Dai, J. Xi, and H.-L. Dai, "Improve cross-project just-in-time defect prediction with dynamic transfer learning," *Journal of Systems and Software*, p. 112214, Sep. 2024.

[75] W. Wang, F. Dumont, N. Niu, and G. Horton, "Detecting software security vulnerabilities via requirements dependency analysis," *IEEE Transactions on Software Engineering*, vol. 48, no. 5, pp. 1665–1675, 2022.

[76] J. Hu, Z. Zhao, F. Hang, and J. Yin, "Effective application of artificial intelligence techniques in security risk assessment and dependency analysis of open source components," *Applied mathematics and nonlinear sciences*, 2024.

[77] RyotaK, "Potential remote code execution in pypi," 2021. [Online]. Available: https://blog.ryotak.net/post/pypi-potential-remote-code-execution-en/

[78] J. Williams, "What you need to know about the solarwinds supply-chain attack — sans institute," 2020. [Online]. Available: https://www.sans.org/blog/what-you-need-to-know-about-the-solarwinds-supply-chain-attack/

[79] O. VLCEK, "Recent findings from ccleaner apt investigation reveal that attackers entered the piriform network via teamviewer," 2018. [Online]. Available: https://blog.avast.com/update-ccleaner-attackers-entered-via-teamviewer

[80] K. Lab, "Operation shadowhammer: new supply chain attack threatens hundreds of thousands of users worldwide," 2019. [Online]. Available: https://www.kaspersky.com/about/press-releases/2019_operation-shadowhammer-new-supply-chain-attack

[81] S. H. Karan Sood, "Notpetya technical analysis – a triple threat: File encryption, mft encryption, credential theft," 2017. [Online]. Available: https://www.crowdstrike.com/blog/petrwrap-ransomware-technical-analysis-triple-threat-file-encryption-mft-encryption-credential-theft/

[82] J. Allen, "Kaseya vsa ransomware attack explained," 2023. [Online]. Available: https://purplesec.us/kaseya-ransomware-attack-explained/

[83] CISA, "Exploitation of accellion file transfer appliance," 2021. [Online]. Available: https://www.cisa.gov/news-events/cybersecurity-advisories/aa21-055a

[84] A. HOPE, "Codecov supply chain attack remained undetected for months and potentially affected major companies including google, ibm, hp, and others," 2021. [Online]. Available: https:

//www.cpomagazine.com/cyber-security/codecov-supply-chain-attack-remained-undetected-for-months-and-potentially-affected-major-companies-including-google-ibm-hp-and-others/

[85] picussecurity, "Tactics, techniques, and procedures (ttps) used by hafnium to target microsoft exchange servers," 2021. [Online]. Available: https://www.picussecurity.com/resource/blog/ttps-hafnium-microsoft-exchange-servers

[86] E. Brumaghin, E. Carter, W. Mercer, M. Olney, P. Rascagneres, and C. Williams, "Ccleaner command and control causes concern," 2017. [Online]. Available: https://blog.talosintelligence.com/ccleaner-c2-concern/

[87] K. Lab, "Operation shadowhammer: new supply chain attack threatens hundreds of thousands of users worldwide," 2019. [Online]. Available: https://www.kaspersky.com/about/press-releases/operation-shadowhammer-new-supply-chain-attack

[88] FireEye, "Highly evasive attacker leverages solarwinds supply chain to compromise multiple global victims with sunburst backdoor," 2020. [Online]. Available: https://www.mandiant.com/resources/blog/evasive-attacker-leverages-solarwinds-supply-chain-compromises-with-sunburst-backdoor

[89] K. Basu, S. M. Saeed, C. Pilato, M. Ashraf, M. T. Nabeel, K. Chakrabarty, and R. Karri, "CAD-Base: An Attack Vector into the Electronics Supply Chain," *ACM Transactions on Design Automation of Electronic Systems*, vol. 24, no. 4, pp. 1–30, 2019.

[90] X. Yi, Y. Fang, D. Wu, and L. Jiang, "BlockScope: Detecting and Investigating Propagated Vulnerabilities in Forked Blockchain Projects," in *Proceedings 2023 Network and Distributed System Security Symposium*. San Diego, CA, USA: Internet Society, 2023.

[91] S. White, "Tajmahal advanced persistent threat," 2019. [Online]. Available: https://www.infoblox.com/wp-content/uploads/threat-intelligence-report-tajmajal-advanced-persistent-threat.pdf

[92] Y. Ren, Y. Xiao, Y. Zhou, Z. Zhang, and Z. Tian, "Cskg4apt: A cybersecurity knowledge graph for advanced persistent threat organization attribution," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 6, pp. 5695–5709, 2023.

[93] P. Gao, F. Shao, X. Liu, X. Xiao, Z. Qin, F. Xu, P. Mittal, S. R. Kulkarni, and D. Song, "Enabling efficient cyber threat hunting with cyber threat intelligence," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, 2021, pp. 193–204.

[94] B. Tang, J. Wang, Z. Yu, B. Chen, W. Ge, J. Yu, and T. Lu, "Advanced persistent threat intelligent profiling technique: A survey," *Computers and Electrical Engineering*, vol. 103, p. 108261, 2022.

[95] R. Alkhadra, J. Abuzaid, M. AlShammari, and N. Mohammad, "Solar Winds Hack: In-Depth Analysis and Countermeasures," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. Kharagpur, India: IEEE, 2021, pp. 1–7.

[96] J. Martínez and J. M. Durán, "Software Supply Chain Attacks, a Threat to Global Cybersecurity: SolarWinds' Case Study," *International Journal of Safety and Security Engineering*, vol. 11, no. 5, pp. 537–545, 2021.

[97] J. P. Farwell and R. Rohozinski, "Stuxnet and the future of cyber war," *Survival*, vol. 53, no. 1, pp. 23–40, 2011.

[98] F. Nicolas, O. M. Liam, and C. Eric, "W32.stuxnet dossier," 2011. [Online]. Available: https://scms-summit.com/media/9417/managing-risks-in-supply-chains-with-digital-twins-and-simulation.pdf

[99] R. Kumar, R. Kela, S. Singh, and R. Trujillo-Rasua, "APT attacks on industrial control systems: A tale of three incidents," *International Journal of Critical Infrastructure Protection*, vol. 37, no. C, 2022.

[100] S. Wrycza and J. Maślankowski, Eds., *Information Systems: Research, Development, Applications, Education: 12th SIGSAND/PLAIS EuroSymposium 2019, Gdansk, Poland, September 19, 2019, Proceedings*, ser. Lecture Notes in Business Information Processing. Cham: Springer International Publishing, 2019, vol. 359.

[101] G. Ho, M. Dhiman, D. Akhawe, V. Paxson, S. Savage, G. M. Voelker, and D. Wagner, "Hopper: Modeling and Detecting Lateral Movement (Extended Report)," 2021.

[102] J. Yang, Q. Zhang, X. Jiang, S. Chen, and F. Yang, "Poirot: Causal Correlation Aided Semantic Analysis for Advanced Persistent Threat Detection," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3546–3563, 2022.

[103] C. Xiong, T. Zhu, W. Dong, L. Ruan, R. Yang, Y. Cheng, Y. Chen, S. Cheng, and X. Chen, "A Practical Real-Time APT Detection System With High Accuracy and Efficiency," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 1, pp. 551–565, 2022.

[104] I. J. King and H. H. Huang, "Euler: Detecting Network Lateral Movement via Scalable Temporal Link Prediction," *ACM Transactions on Privacy and Security*, p. 3588771, 2023.

[105] X. Du, G. Zheng, K. Wang, J. Feng, W. Deng, M. Liu, B. Chen, X. Peng, T. Ma, and Y. Lou, "Vul-rag: Enhancing llm-based vulnerability detection via knowledge-level rag," 2024. [Online]. Available: https://arxiv.org/abs/2406.11147

[106] G. D. Pasquale, I. Grishchenko, R. Iesari, G. Pizarro, L. Cavallaro, C. Kruegel, and G. Vigna, "ChainReactor: Automated privilege escalation chain discovery via AI planning," in *33rd USENIX Security Symposium (USENIX Security 24)*. Philadelphia, PA: USENIX Association, Aug. 2024, pp. 5913–5929.

[107] P. Rieger, T. D. Nguyen, M. Miettinen, and A.-R. Sadeghi, "DeepSight: Mitigating Backdoor Attacks in Federated Learning Through Deep Model Inspection," in *Proceedings 2022 Network and Distributed System Security Symposium*. San Diego, CA, USA: Internet Society, 2022.

[108] S. Cheng, G. Tao, Y. Liu, S. An, X. Xu, S. Feng, G. Shen, K. Zhang, Q. Xu, S. Ma, and X. Zhang, "BEAGLE: Forensics of Deep Learning Backdoor Attack for Better Defense," in *Proceedings 2023 Network and Distributed System Security Symposium*. San Diego, CA, USA: Internet Society, 2023.

[109] F. N. Froh, M. F. Gobbi, and J. Kinder, "Differential static analysis for detecting malicious updates to open source packages," in *Proceedings of the 2023 Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*, ser. SCORED '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 41–49.

[110] R. Liu, Y. Wang, H. Xu, J. Sun, F. Zhang, P. Li, and Z. Guo, "Vul-lmgnns: Fusing language models and online-distilled graph neural networks for code vulnerability detection," *Information Fusion*, p. 102748, 2024.

[111] N. C. K. Yiu, "Toward Blockchain-Enabled Supply Chain Anti-Counterfeiting and Traceability," *Future Internet*, vol. 13, no. 4, pp. 1–26, March 2021.

[112] M. D. Islam, H. Shen, and S. Badsha, "Integrating blockchain into supply chain safeguarded by puf-enabled rfid," *Internet of Things*, vol. 18, p. 100505, 2022.

[113] Y. Ren, Y. Xiao, Y. Zhou, Z. Zhang, and Z. Tian, "Cskg4apt: A cybersecurity knowledge graph for advanced persistent threat organization attribution," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 5695–5709, 2023.

[114] J. Yin, M. Tang, J. Cao, M. You, H. Wang, and M. Alazab, "Knowledge-driven cybersecurity intelligence: Software vulnerability coexploitation behavior discovery," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 4, pp. 5593–5601, 2023.

[115] A. Imran and R. Agrawal, "Data Provenance," in *Encyclopedia of Big Data*, L. A. Schintler and C. L. McNeely, Eds. Cham: Springer International Publishing, 2017, pp. 1–4.

[116] W. Ametepe, C. Wang, S. K. Ocansey, X. Li, and F. Hussain, "Data provenance collection and security in a distributed environment: A survey," *International Journal of Computers and Applications*, vol. 43, no. 1, pp. 11–25, 2021.

[117] R. Shanmugam, "Elements of causal inference: Foundations and learning algorithms," *Journal of Statistical Computation and Simulation*, vol. 88, no. 16, pp. 3248–3248, 2018.

[118] A. R. Nogueira, A. Pugnana, S. Ruggieri, D. Pedreschi, and J. Gama, "Methods and tools for causal discovery and causal inference," *WIREs Data Mining and Knowledge Discovery*, vol. 12, no. 2, 2022.

[119] J. Allen, Z. Yang, M. Landen, R. Bhat, H. Grover, A. Chang, Y. Ji, R. Perdisci, and W. Lee, "Mnemosyne: An Effective and Efficient Postmortem Watering Hole Attack Investigation System," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. Virtual Event USA: ACM, 2020, pp. 787–802.

[120] D. H. Xiong, "Association analysis: Basic concepts and algorithms," 2005. [Online]. Available: https://api.semanticscholar.org/CorpusID:46666286

[121] M. Khosravi and B. T. Ladani, "Alerts correlation and causal analysis for apt based cyber attack detection," *IEEE Access*, 2020.

[122] S. M. Milajerdi, R. Gjomemo, B. Eshete, R. Sekar, and V. Venkatakrishnan, "Holmes: Real-time apt detection through correlation of suspicious information flows," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 1137–1152.

[123] G. Wang, Y. Cui, J. Wang, L. Wu, and G. Hu, "1. a novel method for detecting advanced persistent threat attack based on belief rule base," *Applied Sciences*, 2021.

[124] G. Yan, Q. Li, D. Guo, and X. Meng, "Discovering Suspicious APT Behaviors by Analyzing DNS Activities," *Sensors*, vol. 20, no. 3, p. 731, 2020.

[125] Z. A. El Houda, A. S. Hafid, and L. Khoukhi, "A Novel Machine Learning Framework for Advanced Attack Detection using SDN,"

in *2021 IEEE Global Communications Conference (GLOBECOM)*. Madrid, Spain: IEEE, 2021, pp. 1–6.

[126] Y. Shen, M. Simsek, B. Kantarci, H. T. Mouftah, M. Bagheri, and P. Djukic, "Prior Knowledge based Advanced Persistent Threats Detection for IoT in a Realistic Benchmark," in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*. Rio de Janeiro, Brazil: IEEE, 2022, pp. 3551–3556.

[127] M. Naseri, Y. Han, E. Mariconti, Y. Shen, G. Stringhini, and E. D. Cristofaro, "Cerberus: Exploring federated prediction of security events," 2022.

[128] P. Gohel, P. Singh, and M. Mohanty, "Explainable AI: Current status and future directions," 2021.

[129] B. Wang, T. Zhou, S. Li, Y. Cao, and N. Gong, "GraphTrack: A Graph-based Cross-Device Tracking Framework," in *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*. Nagasaki Japan: ACM, 2022, pp. 82–96.

[130] K. Sood, "Protecting the software supply chain: Deep insights into the ccleaner backdoor," 2017. [Online]. Available: https://www.crowdstrike.com/blog/protecting-software-supply-chain-deep-insights-ccleaner-backdoor/

[131] K. T. Francis Guibernau, "Emulating recent activity from the russian adversary nobelium / apt29," 2023. [Online]. Available: https://www.attackiq.com/2023/05/04/emulating-nobelium-apt29/

[132] J. W. Francis Guibernau, Ken Towne, "Oilrig attack graphs: Emulating the iranian threat actor's global campaigns," 2022. [Online]. Available: https://www.attackiq.com/2022/07/11/oilrig-attack-graphs-emulating-the-iranian-threat-actors-global-campaigns/

[133] Kaspersky, "Advanced persistent threat actor lazarus attacks defense industry, develops supply chain attack capabilities," 2021. [Online]. Available: https://www.kaspersky.com/about/press-releases/2021_advanced-persistent-threat-actor-lazarus-attacks-defense-industry-develops-supply-chain-attack-capabilities

[134] Y. Zheng, S. Pujar, B. Lewis, L. Buratti, E. Epstein, B. Yang, J. Laredo, A. Morari, and Z. Su, "D2a: a dataset built for ai-based vulnerability detection methods using differential analysis," in *Proceedings of the 43rd International Conference on Software Engineering: Software Engineering in Practice*, ser. ICSE-SEIP '21. IEEE Press, 2021, p. 111–120.

[135] P. Wang, S. Liu, L. Ai, and W. Jiang, "Detecting security vulnerabilities with vulnerability nets," *Journal of Systems and Software*, vol. 208, pp. 111 902–111 902, 2024.

[136] X. Xu, Q. Wang, H. Li, N. Borisov, C. A. Gunter, and B. Li, "Detecting ai trojans using meta neural analysis," *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 103–120, 2019.

[137] L.-X. Yang, P. Li, X. Yang, and Y. Y. Tang, "A Risk Management Approach to Defending Against the Advanced Persistent Threat," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 6, pp. 1163–1172, 2020.

[138] X. Guo, Z.-Y. Yang, J. Sun, and Y. Zhang, "Impact pathways of emerging its to mitigate supply chain vulnerability: A novel dematel-ism approach based on grounded theory," *Expert Systems With Applications*, 2023.

[139] U. Agarwal, V. Rishiwal, S. Tanwar, R. Chaudhary, G. Sharma, P. N. Bokoro, and R. Sharma, "Blockchain technology for secure supply chain management: A comprehensive review," *IEEE Access*, vol. 10, pp. 85 493–85 517, 2022.

[140] E. Bandara, S. Shetty, A. Rahman, and R. Mukkamala, "Let'sTrace — Blockchain, Federated Learning and TUF/In-ToTo Enabled Cyber Supply Chain Provenance Platform," in *MILCOM 2021 - 2021 IEEE Military Communications Conference (MILCOM)*. San Diego, CA, USA: IEEE, 2021, pp. 470–476.

[141] S. T. Liu, "Counting processes and survival analysis," *Technometrics*, vol. 49, no. 3, pp. 362–362, 2007.

[142] L. D. Fisher and D. Y. Lin, "Time-dependent covariates in the cox proportional-hazards regression model," *Annual review of public health*, vol. 20, no. 1, pp. 145–157, 1999.

[143] R. J. Rodríguez, X. Chang, X. Li, and K. S. Trivedi, "Survivability analysis of a computer system under an advanced persistent threat attack," in *Graphical Models for Security*, B. Kordy, M. Ekstedt, and D. S. Kim, Eds. Cham: Springer International Publishing, 2016, pp. 134–149.

[144] I. Ghafir, K. G. Kyriakopoulos, S. Lambotharan, F. J. Aparicio-Navarro, B. AsSadhan, H. Binsalleeh, and D. M. Diab, "Hidden markov models and alert correlations for the prediction of advanced persistent threats," *IEEE Access*, vol. 7, pp. 99 508–99 520, 2019.

[145] P. Goovaerts, "Comparative performance of indicator algorithms for modeling conditional probability distribution functions," *Mathematical Geology*, vol. 26, pp. 389–411, 1994.

[146] A. Tsoularis and J. Wallace, "Analysis of logistic growth models," *Mathematical biosciences*, vol. 179, no. 1, pp. 21–55, 2002.

[147] X. Han, T. Pasquier, A. Bates, J. Mickens, and M. Seltzer, "Unicorn: Runtime provenance-based detector for advanced persistent threats." The Internet Society, 2020, pp. 1–18.

[148] R. Arantes, C. Weir, H. Hannon, and M. Kulseng, "Operationally transparent cyber (optc)," 2021. [Online]. Available: https://dx.doi.org/10.21227/edq8-nk52

[149] J. Reha, G. Lovisotto, M. Russo, A. Gravina, and C. Grohnfeldt, "Anomaly detection in continuous-time temporal provenance graphs," in *Temporal Graph Learning Workshop @ NeurIPS 2023*, 2023.

[150] A. Salem, M. Backes, and Y. Zhang, "Get a Model! Model Hijacking Attack Against Machine Learning Models," in *Proceedings 2022 Network and Distributed System Security Symposium*. San Diego, CA, USA: Internet Society, 2022.

[151] X. Lin, X. Liu, B. Chen, Y. Wang, C. Dong, and P. Hu, "Atal: Active learning using adversarial training for data augmentation," *IEEE Internet of Things Journal*, vol. 11, pp. 4787–4800, 2024.

[152] H. Eghbal-zadeh, W. Zellinger, M. Pintor, K. Grosse, K. Koutini, B. A. Moser, B. Biggio, and G. Widmer, "Rethinking data augmentation for adversarial robustness," *Information Sciences*, 2024.

[153] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021.

[154] R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu, "Embracing change: Continual learning in deep neural networks," *Trends in Cognitive Sciences*, vol. 24, pp. 1028–1040, 2020.

[155] R. Cantini, A. J. Orsino, and D. Talia, "Xai-driven knowledge distillation of large language models for efficient deployment on low-resource devices," *Journal of Big Data*, vol. 11, pp. 1–17, 2024.

[156] Z. Zeng, Q. Peng, X. Mou, and Y. Wan, "Graph neural networks with high-order polynomial spectral filters." *IEEE transactions on neural networks and learning systems*, vol. PP, 2023.

[157] C. Cimpanu, "Eset discovers a rare apt that stayed undetected for nine years," 2020. [Online]. Available: https://www.zdnet.com/article/eset-discovers-a-rare-apt-that-stayed-undetected-for-nine-years/

[158] T. Hegel, "Modifiedelephant apt and a decade of fabricating evidence," 2022. [Online]. Available: https://www.sentinelone.com/labs/modifiedelephant-apt-and-a-decade-of-fabricating-evidence/

[159] S. Khandelwal, "Sophisticated 'tajmahal apt framework' remained undetected for 5 years," 2019. [Online]. Available: https://thehackernews.com/2019/04/apt-malware-framework.html

[160] D. MCWHORTER, "Apt28: A window into russia's cyber espionage operations?" 2014. [Online]. Available: https://www.mandiant.com/resources/blog/apt28-a-window-into-russias-cyber-espionage-operations

[161] Kaspersky, "Equation group: The crown creator of cyber-espionage," 2021. [Online]. Available: https://www.kaspersky.com/about/press-releases/2015_equation-group-the-crown-creator-of-cyber-espionage

[162] Mandiant, "Equation group: The crown creator of cyber-espionage," 2021. [Online]. Available: https://www.mandiant.com/sites/default/files/2021-09/mandiant-apt1-report.pdf

[163] S. Doherty, "Hidden lynx – professional hackers for hire," 2013. [Online]. Available: https://www.wired.com/images_blogs/threatlevel/2013/09/hidden_lynx_final.pdf

[164] Kaspersky, "Kaspersky lab analyzes active cyberespionage campaign targeting online gaming companies worldwide," 2013. [Online]. Available: https://www.kaspersky.com/about/press-releases/2013_kaspersky-lab-analyzes-active-cyberespionage-campaign-targeting-online-gaming-companies-worldwide

[165] J. Fokker, "Suspected darkhotel apt activity update," 2022. [Online]. Available: https://www.trellix.com/en-us/about/newsroom/stories/research/suspected-darkhotel-apt-activity-update.html

[166] S. D. Luis Magisa, "New macos backdoor connected to oceanlotus surfaces," 2020. [Online]. Available: https://www.trendmicro.com/en_us/research/20/k/new-macos-backdoor-connected-to-oceanlotus-surfaces.html

[167] A. Drozhzhin, "Russian-speaking cyber spies exploit satellites," 2015. [Online]. Available: https://www.kaspersky.co.uk/blog/turla-apt-exploiting-satellites/6210/

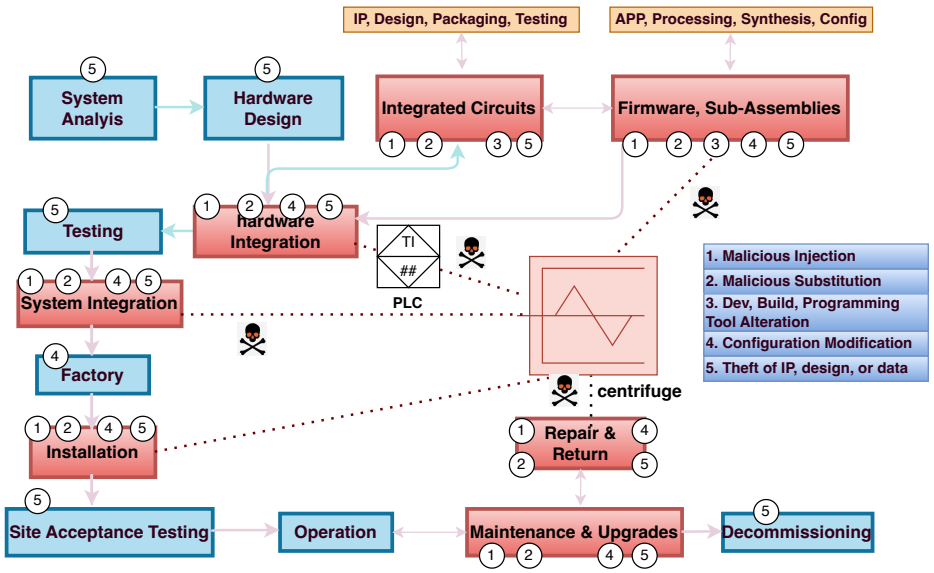

Fig. 8: Nuclear Power Attack Scenario

$\cdots$ denotes severe impact, $\rightarrow$ denotes one-way data flow, $\leftrightarrow$ denotes bidirectional data flow

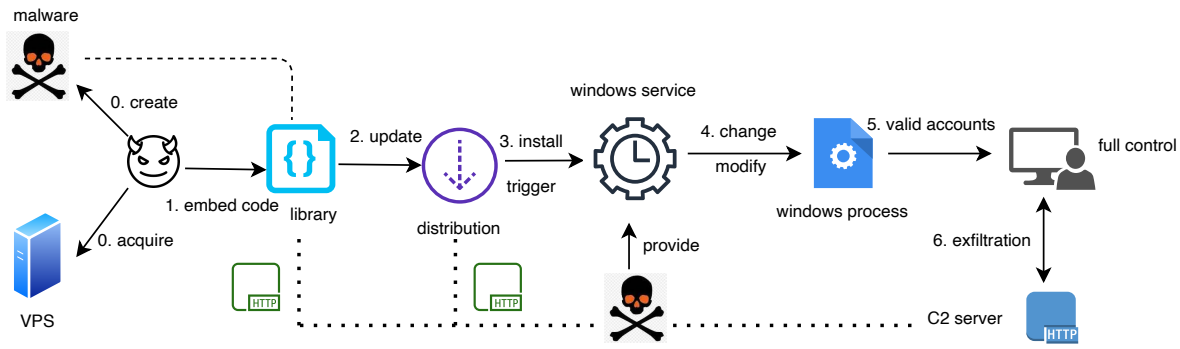

Fig. 9: SolarWinds Attack Scenario

Appendix

Supplementary Table for Undetected APT Attacks Investigation

Supplementary Diagram for Nuclear Power Attack Scenario

Supplementary Diagram for SolarWinds Attack Scenario

Supplementary Table for Comparative Analysis of Supply Chain Security Solutions

TABLE VII: Undetected APT Attacks Investigation

Name	Techniques	Initial Stage	Communication	Supply Chain
XDSpy [157]	Exploitation for Client Execution, Office Application Startup: Outlook, Develop Capabilities: Malware, Active Scanning, Data from Local System, Gather Victim Identity Information	Spearphishing Link, Spearphishing via Service, Watering Hole (WH)	HTTPS	No
Modified Elephant [158]	Develop Capabilities: Malware, Develop Capabilities: Digital Certificates, Input Capture: Keylogging, Exploit Public-Facing Application, Valid Accounts: Cloud Accounts, Automated Exfiltration, Remote Access Software	Spearphishing Link, Spearphishing via Service	Develop Capabilities: Malware	No
Cozy Bear [131]	Local Account, Brute Force: Credential Stuffing, Steal Web Session Cookie, Windows Management Instrumentation, Additional Cloud Credentials, Office Application Startup: Outlook, Inter-Process Communication: Pass Access Token, Logon Session: Pass the Hash, Exploit Public-Facing Application, Scheduled Task/Job: Scheduled Task	Phishing, WH	Web Services: Exfiltration Over Asymmetric Encrypted Non-C2 Protocol, Custom Protocols (CPS)	Yes
TajMahal [159]	Pre-OS Boot: Component Firmware, Process Injection: Dynamic-link Library Injection, Input Capture: Screen Capture, Archive Collected Data, Data Staged: Archive via Utility, Upload Malware, Tool, Credential Dumping: Credential API Hooking, Input Capture: Keylogging, System Profiling, Video Capture, SDM, Self-Updating Mechanisms,	Phishing	Custom DNS Protocol, HB, HTTPS	Maybe
Fancy Bear [160]	Phishing, WH, DT, RE Zero-Day Vulnerabilities (ZDV), Input Capture: Keylogging, Data Staged: Archive via Utility	Gather Victim Identity Information	Domain Fronting (DF), Encrypted Channel: Asymmetric Cryptography	Yes
Lazarus Group [133]	WH, Phishing, ZDV, Defacement, Remote Command Execution, Input Capture: Keylogging, Data Staged: Archive via Utility	Gather Victim Host Information, Gather Victim Network Information, Gather Victim Identity Information	Proxy, Encrypted Channel: Asymmetric Cryptography, Upload Malware	Yes
Equation Group [161]	ZDV, Rootkit, Resource Enhancement (RE), Pre-OS Boot: Bootkit, DE Network Protocol Interference, Command Execution (CE), Installing Payloads ,HB	Gather Victim Host Information, Gather Victim Network Information, Active Scanning	CPS, Data Obfuscation: Junk Data	Yes
Comment Crew [162]	ZDV, WH, HB, Remote Access Software Phishing, Exploitation for Privilege Escalation	Gather Victim Host Information, Gather Victim Network Information, Active Scanning, Gather Victim Identity Information	Exploit Public-Facing Application: DNS Server, Encrypted Channel: Asymmetric Cryptography, Web Services: Exfiltration Over Asymmetric Encrypted Non-C2 Protocol	No
Hidden Lynx [163]	Phishing, WH, ZDV, Remote Access Software, Input Capture: Keylogging, Data Staged: Archive via Utility	Gather Victim Host Information, Gather Victim Network Information, Active Scanning, Gather Victim Identity Information, Phishing	Data Obfuscation: Steganography, Encrypted Channel: Asymmetric Cryptography	No
Winnti Group [164]	Phishing, WH, Exploit Public-Facing Application: Web Server, HB, Input Capture: Keylogging, Remote Access Software	Active Scanning for Exposed Credentials	Web Services: Exfiltration Over Asymmetric Encrypted Non-C2 Protocol, Proxy, HTTPS, Upload Malware: Content Upload, CPS	Yes
DarkHotel [165]	WH, Phishing, Remote Access Software Input Capture: Keylogging	Gather Victim Identity Information, Gather Victim Organizations	Wi-Fi Evil Twin, Phishing, Upload Malware: Content Upload	No
OceanLotus [166]	WH, Phishing, HB, Valid Accounts: Cloud Accounts, Input Capture: Keylogging, Remote Access Software	Gather Victim Host Information, Gather Victim Network Information, Phishing	HB, Web Services: Exfiltration Over Asymmetric Encrypted Non-C2 Protocol	No
Turla Group [167]	Hijack Update (HU), HB, WH, Data Manipulation, Rootkit, DT, Input Capture: Keylogging, Data Staged: Archive via Utility	Gather Victim Network Information, Gather Victim Identity Information	Upload Malware: Content Upload, Proxy: External Proxy, Satellite-Based Communication Channel	Yes

TABLE VIII: Comparative Analysis of Supply Chain Security Solutions

Solutions	Core Method	Scalability	Dataset Size	F1-Score	Compute Overhead	Focused Stage	Generality	Drawbacks
Pasq. et al. [72] (2024)	Transfer Learning	low	17000	82.9%	-	source code	low	lack of diversity in dataset
Dai et al. [74] (2024)	CPG	medium	207697	53.4%	high	source code	high	dependence on historical data; heterogeneity of data
Rozi et al. [73] (2024)	CPG, GCN	low	5578	70.98%	-	source code	low	sensitive to unknown words / vulns; struggle to capture crucial textual info
Ladisa et al. [71] (2023)	ML	high	194	84.4%	low	source code	medium	sample scarcity; no benchmark comparison
Nata. et al. [69] (2024)	Blockchain, Cloud	high	-	-	adaptive	transform, integrity	high	-
Khan et al. [68] (2023)	Federated Learning	high	22339021	99.32%	-	transform; privacy	uncertain	require live data to evaluate
Tran et al. [66] (2024)	BERT, GNN	medium	211317	74.10%	high	source code	high	high space complexity
Lu et al. [67] (2024)	LLM	high	22361; 179299; 18169	66.38%, 35.93%, 42.26%	high	source code	high	reliance on the scale of training data; extensive training time
Wang et al. [75] (2022)	Dependency Analysis	medium	-	73%	high	dependency	high	required manual analysis; reliance on indicators
Hu et al. [76] (2024)	Reinforcement Learning	high	10987	97.18%	-	dependency	high	-
Cerch. et al. [70] (2023)	Blockchain	high	-	-	high	transform, privacy	medium	trust issue; regulatory and legal aspects