CYBERDEFENSE REPORT

Software Supply Chain Attacks
An Illustrated Typological Review

Sean Cordey

Zürich, January 2023
Center for Security Studies (CSS), ETH Zürich

CSS
ETH Zürich

ETH zürich

Available online at: https://css.ethz.ch/en/publications/risk-and-resilience-reports.html

Author: Sean Cordey

ETH-CSS project management:
Stefan Soesanto, Project Lead Cyberdefense;
Andrin Hauri, Head of the Risk and Resilience Team;
Andreas Wenger, Director of the CSS.

Editor: Jakob Bund, Benjamin Scharte, Stefan Soesanto, Taylor Grossman
Layout and graphics: Miriam Dahinden-Ganzoni

DOI: 10.3929/ethz-b-000584947

Table of Content

1 Introduction

1.1 Supply Chain Attacks in the Public Discourse

Most elements constituting modern life, from the economy to social habits, are now characterized by using digital technologies and the consumption of goods and services that depend on complex, interconnected, transnational, and, at times, vulnerable, supply chains. Critical dependencies and heightened (cyber) threats combined with strategic competitiveness are increasingly turning the issue of supply chain security into matters of national and international security.

Supply chain security is generally defined as the security of the ecosystem of processes, people, organizations, and distributors involved in the development, manufacturing, and delivery of finished solutions or products. Over the past few years, the discussions surrounding the importance of supply chains in the context of the national security and stability of the global economy have been multifaceted and often reactive to current events. During the pandemic, the public discussions surrounding supply chains have largely revolved around workforce disruptions, maritime bottlenecks, surging energy prices, and the resulting price increases and shortages of goods. Prior to the pandemic the focus was instead on the supply of technological goods (e.g., 5G infrastructure), superconductors and chip components, as well as strategic materials and rare minerals. All the while, recent cyber incidents (e.g., SolarWinds, Sunburst, JBS, Colonial Pipeline) have thrown the vulnerability of supply chains to cyber risks into sharp relief across different industries.

Located at the intersection of supply chains and cyber are the topics of software supply chains attacks and broader mitigation and protection elements that fall under the term cyber supply chain risk management (C-SCRM)[1]. Due to their heightened relevance in the current security discourse, their potential destructive and strategic effects, and their increased use by malicious actors (state-linked and criminal), software supply chain attacks will be the focus of this report.

Supply chain attacks are not a new issue per se. Software supply chain attacks can be traced as far back as 1974, when a tiger team in the US Air Force penetrated the MIT's Multics time-sharing operating system and inserted a "trap door" that made it into Honeywell's master copy prior to distribution.[2] Despite its frequent use, the term "supply chain attacks" – as with most emerging and operational terms – is still surrounded by a great deal of fuzziness as the building blocks and various assumptions as to what supply chain attacks actually consist of remains diffuse. For example, supply chain attack techniques have been described in great variety ranging from planting backdoors in firmware, hijacking third-party updating mechanisms, exploiting trusted relationships by undermining code-signing, and even compromising open-source code used in software development.

Despite this cacophony of different and partially overlapping typologies, several definitions have been published by government agencies, institutions, and think tanks. These definitions largely distinguish themselves by emphasizing different elements and components within a supply chain attack, such as techniques, targeted elements, or their lifecycle. Additionally, several of these definitions are part of – or related to – other closed technical standards, guidance, or best practices related to C-SCRM. As such, they are not always accessible to the broader public and policymakers, who might not have a vested interest in the subject matter.

[1] See for instance, the NIST framework or the Securing the Software Supply Chain: Recommended Practices for Suppliers and Developers reports. New or upcoming regulation also have such elements, such as the GDPR Directive on Supply Chain Risk or NIS2 directive.

[2] Paul A. Karger & Roger R. Schell, "Multics Security Evaluation: Vulnerability Analysis," June 1974, https://seclab.cs.ucdavis.edu/projects/history/papers/karg74.pdf, p. 50-54. According to Daniel Miessler's research, a tiger team was defined in a 1964 paper as "'a team of undomesticated and uninhibited technical specialists, selected for their experience, energy, and imagination, and assigned to track down relentlessly every possible source of failure in a spacecraft subsystem.' The term is now used often as a synonym for Red Team, but the general definition is an elite group of people designed to solve a particular technical challenge." See: Daniel Miessler, "The Difference Between Red, Blue, and Purple Teams," August 12, 2021, https://danielmiessler.com/study/red-blue-purple-teams/.

1.2 Structure & Aim

The overarching aim of this report is to provide an illustrative overview of software supply chain attacks (SSCA) and to raise awareness of the types of attacks, their uses, and their potential impacts.

To set the foundation, section two starts with a terminology analysis, which also explains the main properties underpinning the concepts of supply chains, software supply chains, and software supply chain attacks. Section three focuses on reviewing the different threat frameworks that address and describe SSCAs. This includes detailing the diverse set of actors, impacts, and assets involved in SSCAs. The section also illustrates the large variety of threat vectors and techniques that can be used across a software's lifecycle to conduct an SSCA. Section four describes the underlying assumptions, enabling factors, and trends behind SSCAs. More precisely, it includes an overview of the professed operational advantages of SSCAs, the messy software ecosystem making SSCAs viable and hard to detect, and the changes in adversarial behavior to leverage SSCAs. The final section explores the larger spectrum and implications behind software supply chain attacks. The first sub-section examines SSCA edge cases (i.e., cases that are not typically viewed as being SSCAs), those that are falsely classified as SSCAs, and those that are novel forms or present unique features of SSCAs. The report concludes with a final sub-section that looks at the potential policy and practical implications of precisely differentiating, classifying, and characterizing certain types of cyberattacks.

2 Definitions and Typological Conundrums

The starting point of this report is an analysis of the terminology in use. This requires defining three essential terms: supply chains, cyber/information and communication technology (ICT) supply chains, and (software) supply chain attacks. To do so, an array of publications have been reviewed and summarized (see Annex). This literature review includes publications from academic institutions and think tanks, national and international ICT standards institutions, national cybersecurity centers, national Computer Emergency Response Teams (CERT), and threat intelligence companies.

2.1 Shades of Supply Chains

2.1.1 Supply Chain

As presented in in the Annex, the term supply chain has received several general definitions. For instance, the European Union Agency for Cybersecurity (ENISA) defines it as "a system of organizations, people, technology, activities, information and resources involved in moving a product or service from supplier (producer) to customer."[3] By contrast, the American National Institute of Standards and Technology (NIST) views it as a "[l]inked set of resources and processes between and among multiple tiers of developers, each of which is an acquirer, that begins with the sourcing of products and services and extends through their life cycle."[4] While these two definitions are markedly different, they nonetheless overlap considerably and offer a set of key constitutive elements:

- A system/ecosystem/ network of people, technologies, information, resources, organizations, distributors, activities, or operations.
- A set of relationships that can be successive or tiered across multiple levels of enterprises.

[3] ENISA, "Supply Chain Integrity: An Overview of the ICT Supply Chain Risks and Challenges, and Vision for the Way Forward," September 2015, p. 7, https://www.enisa.europa.eu/publications/sci-2015/@@download/fullReport.

[4] NIST, "Security and Privacy Controls for Information Systems and Organizations," Special Publication 800-53 Rev.5, p.419, https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final.

- A set of agents that traditionally includes (but is not limited to) vendors, manufacturing facilities, logistics providers, distribution centers, distributors, wholesalers, retailers, integrators, and suppliers.
- A set of operations/activities throughout a product or service life cycle that can include (but is not limited to) sourcing, handling, developing, manufacturing, transforming, processing, distributing, and delivering raw materials and components.

The totality of these elements is structured to ship a product or service from a supplier to a customer (either final or not).

2.1.2 Cyber Supply Chain

Increasingly used across different policy spheres, the terms ICT/cyber/digital and e-supply chains can be understood as a subset of supply chains, with a specific focus on the development and use of ICT products and services. NIST, alongside the US Cybersecurity and Infrastructure Security Agency (CISA), define it as: "a [l]inked set of resources and processes between acquirers, integrators, and suppliers that begins with the design of ICT products and services and extends through development, sourcing, manufacturing, handling, and delivery of ICT products and services to the acquirer."[5]

Compared to the general definition of supply chains, the definitions for cyber supply chains are even more diverse (see Annex). This is most likely due to the novel nature of the elements being described, as well as the sources and intents behind these different definitions at any given point in time. It is nonetheless possible to identify a common set of constitutive elements:

- A wider set of covered elements, which include hardware, software, cloud or local storage, distribution mechanism, management application, data, and algorithms.
- The explicit consideration of these components' cybersecurity risks and the defense of information technology (IT) and operational technology (OT) infrastructure.
- The use of ICTs to perform and support value-adding activities.

2.1.3 Software Supply Chain

As a sub-set of supply chains and cyber supply chains, the term software supply chain has been defined in various publications as well (see Annex). An illustrative definition is provided by Barabanov et al., who define the term as: "A system of its participants with an interconnected set of resources and processes involved in the life cycle of software movement from the developer to the end user, namely, design, development, manufacturing, supply, implementation, support of programs and associated services." [6]

Barabanov et al. summarized the key characteristics of a software supply chain as following:

- The overarching goal of delivering a software product or service to end users (i.e., on a Platform-as-a-Service or Software-as-a-Service basis).
- A complex set of relationships between different organizations, such as developers, logistic centers, and distribution and assembly centers, in which each actor in the chain can operate as a supplier and/or a customer.
- The existence of two material/service streams. The upstream connects with the product creation using third-party components. The downstream is associated with the product delivery to the end user through a distribution network. These intricate systems of interaction increase the risk of impaired transparency as the end user has little insights into the quality of the delivered products and services across the entire supply chain.

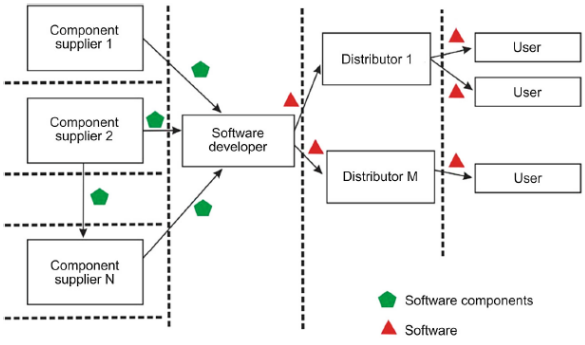

Figure 1: Illustration of the structure of a typical software supply chain (Barabanov et al., 2020)

[5] CISA, "Information and Communications Technology Supply Chain Risk Management Task Force Year 2 Report," December 2020, p. 36, https://www.cisa.gov/sites/default/files/publications/ict-scrm-task-force_year-two-report_508.pdf.

[6] Barabanov et al., "On Systematics of the Information Security of Software Supply Chains," In: Software Engineering Perspectives in Intelligent Systems, Springer, December 2020, pp. 115-129, https://link.springer.com/chapter/10.1007/978-3-030-63322-6_9.

2.2 Software Supply Chain Attacks

The previous definitional overview leads us to the specific interest of this report: software supply chain attacks, or SSCAs for short.

ENISA defines SSCA as "the compromise of a particular asset, e.g. a software provider's infrastructure and commercial software, with the aim to indirectly damage a certain target or targets, e.g. the software provider's clients."[7] Similarly, the MITRE Corporation views it as: "an intentional malicious action (e.g., insertion, substitution or modification) taken to create and ultimately exploit a vulnerability in Information and Communication Technology (hardware, software, firmware) at any point within the supply chain with the primary goal of disrupting or surveilling a mission using cyber resources."[8] Meanwhile, in a report from the Atlantic Council, Trey Herr et al define the term as "when an attacker accesses and edits software in the complex software development supply chain to compromise a target farther up on the chain by inserting their own malicious code."[9]

As with the previous terms, we can also delineate a few key overarching elements that characterize SSCAs:

- The goals behind such attacks can be diverse, ranging from damaging, disrupting, infiltrating, surveilling, and manipulating information, data, or systems at each link of the chain.
- There are numerous potential attack vectors and techniques, including (but not limited to) insider threats, prepositioning by installing backdoors to inject malicious code, and exploiting update mechanisms.
- The malicious exploitation of trust, either between a customer and a third-party supplier, in a software, or within a process (i.e., software updates).
- The impact varies, including loss of confidentiality, data integrity, or the availability of information or information systems. It can also impact an organization's mission, function, image, and reputation; the organizational assets and individuals; and other organizations, up to impacting a nation's entire ecosystem.

To note, none of the definitions outlined capture all SSCA elements within a single framework. Certain organizations prefer to emphasize one over the other due to institutional interests, historical path dependencies, the audience they are talking to, and the in-house knowledge base. For instance, it could be argued that technical institutions like NIST and MITRE are viewing SSCAs in largely technical terms. Meanwhile, the US Committee on National Security Systems – which provides a forum for discussions on national cybersecurity policies – focuses on cyber supply chain risk taxonomies and attack vectors. Others use more policy-oriented definitions that underline the aims and effects of SSCAs.

2.3 The Messy Truth Behind SSCAs

This sub-chapter provides a broad overview on what software supply chain attacks are in practice. Through several examples, it addresses the following questions:

- What type of actors might leverage SSCAs and for what purposes?
- What type of frameworks depict SSCA tactics, techniques, and procedures (TTPs)?
- What TTPs might be leveraged throughout the software lifecycle?
- What type of assets might be targeted by SSCAs?
- And what is the potential impact of SSCAs?

2.4 Threat Actors and Objectives

SSCAs have been leverage by a variety of threat actors, ranging from foreign intelligence services, militaries, corporate spies, corrupt government officials, cyber vandals, disgruntled employees, radical hacktivists, purveyors of counterfeit goods, and others. Two types of

[7] European Union Agency for Cybersecurity, "Supply chain attacks," August 29, 2017, https://www.enisa.europa.eu/publications/info-notes/supply-chain-attacks.

[8] William Heinbockel et al. "Supply Chain Attacks and Resiliency Mitigations – Guidance for System Security Engineers," October 2017, https://www.mitre.org/sites/default/files/2021-11/pr-18-0854-supply-chain-cyber-resiliency-mitigations.pdf

[9] Trey Herr et al., "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 25, 2020, p. 9, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

actors ought to be highlighted, as they remain the most active on SSCAs: Advanced Persistent Threat actors (APTs) and cybercriminal groups.

According to open-source reporting, APTs have been among the main perpetrators of SSCAs over the past decade. Compiling a dataset of over 100+ SSCAs conducted between January 2010 and June 2021, the Atlantic Council assessed that at least 33 were conducted by APTs. These include SSCAs leveraged by APTs from Russia, China, North Korea, Iran, as well as India, Egypt, the United States, and Vietnam. The dataset also underlines that SSCAs are particularly popular amongst Russian and Chinese APTs, and that other countries are quickly catching up by developing their own SSCA capabilities.[10]

Similarly, a more focused dataset published by the European Union Agency for Cybersecurity (ENISA) that covers the period between January 2020 to early July 2021 shows that more than 50% (14 out of 24 SSCAs) were attributable to APT groups.[11] ENISA's finding might suggest a potential future increase in the use of SSCAs by state and state-sponsored actors. Table 1 provides a few examples of different SSCA campaigns and their public attribution to specific threat actors.

As for why APTs are seemingly more prone to leverage SSCAs, the general argument in the literature posits that SSCAs require a certain level of sophistication, resources, patience, and dedication that primarily well-organized and well-funded threat actors can muster. However, as will be discussed later, this assumption needs to be examined. Not all SSCAs are complex or sophisticated. Some SSCAs merely require stolen credentials or simple social engineering techniques.

Incident name	Year	Attributed group
MS Exchange	2021	Hafnium, APT 31 / APT 40
Mimecast	2021	APT29
MoonPass	2021	Winnti APT
Myanmar Presidential Website	2021	Mustang Panda APT
SITA	2021	APT41
Stock Investment Messenger	2021	Thallium APT
SYNNEX	2021	APT29
Azure/ Cloud service providers	2021	Nobelium
NetBeans Project	2020	Octopus Scanner
Able Desktop	2020	TA428
Accellion	2020	UNC2546
Camero	2020	SideWing
SolarWinds Sunburst	2020	APT29
Vietnam VGCA	2020	TA413, TA428
Witvera Veraport	2020	Lazarus APT
Infestation PointBlank	2019	APT41
Sandworm android attack	2019	Sandworm
ShadowHammer	2019	Barium APT
PhantomLance	2018	APT 32 / Ocean Lotus
CCleaner	2017	APT41 or APT17
Equifax data breach	2017	Chinese PLA
KingSlayer	2017	APT31, APT19
NotPetya	2017	Sandworm
ShadowPad	2017	Winnti APT
Soft Cell	2017	APT10/Soft Cell
NetSarang	2017	APT17
DragonFly 2.0 Energy Attack	2015	DragonFly 2.0
EquationDrug and GrayFish	2015	Equation Group
Instart Logic	2015	Syrian Electronic Army
Ukrainian Power Grid attack	2015	Sandworm
Duqu 2.0	2014	Unit 8200
Havex	2014	Russian APT
Flame	2012	Unit 8200
Duqu 1.0	2011	Unit 8200

Table 1: Examples of various SSCAs attributed to APTs.

As for most cyberattacks and activity in this space, deciphering the exact motivation behind a specific SSCA can be relatively arduous. State-linked malicious actors have been found to leverage SSCAs for a wide-array of objectives: strategic and political espionage, economic espionage, disruption, sabotage, and surveillance. By contrast, cybercriminal groups mainly pursue SSCAs for financial gain.

2.4.1 Strategic Espionage

A main objective behind SSCA is cyber espionage. The SolarWinds/Sunburst campaign is a prominent recent example that has grabbed international headlines, but there have been numerous other interesting cases over the years. For instance, in 2018, security researchers found two separate instances of fake Android apps that conducted highly targeted espionage campaigns against

[10] Kurt Baumgartner, "Time to make the Doughnuts," PowerPoint presentation at the Centre for Cybersecurity Belgium (CCB) - Quarterly Cyber Threat Report Event (QCTR) - 2021-Q3.

[11] European Union Agency for Cybersecurity, "ENISA Threat Landscape for Supply chain attacks," July 2021, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport.

targets in the Middle East.[12] Each app spied on around a thousand users in Palestine. The spyware was able to evade Google's security measures by commencing a malicious download after the user had installed and interacted with the app.

The campaign was attributed to APT-C-23, which is thought to be linked to Hamas. Hamas' activity in this space is not surprising. SSCAs are particularly well-suited for intelligence operations. They offer a wide range of stealthy capabilities suited for different operational needs, including data collection and data alteration.[13]

SSCAs also provide an adversary with some avenue for persistence (or persistent access). Indeed, SSCAs have often been used to gain an initial foothold in a targeted system before conducting a series of other intrusions. The 2017 CCleaner incident is particularly interesting in this context. After gaining initial access through a compromised software update, the malicious actors were able to install a backdoor on the systems of approximately 2.2 million CCleaner customers.[14] This allowed the attackers to further attempt to penetrate the networks of 18 companies, including ASUS, Fujitsu, Intel, O2, Singtel, Sony, and VMware to name a few. Persistence is particularly prevalent when an SSCA can gain access to the base source code of a software product to insert a backdoor or create a deliberate vulnerability to exploit. Indeed, once the compromised software product has been published and distributed across the user base, these backdoors and vulnerabilities are likely to persist even when patches and quick fixes are made available.

SSCAs can also remain undetected for quite an extended period of time. In the case of the SolarWinds/Sunburst campaign, the malicious actor was able to operate undetected for eight months before the cyber threat intelligence community and the institutions affected detected it. Some SSCAs, such as APT10's Operation Cloud Hopper, remained undetected for several years.

The situation is even more concerning for SSCAs that target and leverage open-source libraries and registries. According to Duan et al., 20% of the malware introduced into package managers – such as NPM, PyPI, and RubyGems, which allow software developers to easily reuse third party code and simplify the building process – persists for more than 400 days before it is detected and removed.[15] A popular argument that likely explains these long detection times is the lack of consistent and efficient mechanisms that check for malicious code injections in packages that are uploaded into various repositories.[16] This issue may slowly improve as companies have begun to implement new initiatives to curtail these supply chain risks particularly in the aftermath of the SolarWinds/Sunburst revelations. For example, the software development, hosting, and version control platform GitHub has implemented Two Factor Verification (2FA) and is aiming to automatically detect security vulnerabilities and other weaknesses in open-source projects.[17] Meanwhile, the open-source project Sigstore is trying to make it easier for developers to sign their code version releases and for others to verify them – thus overcoming the often absent "code signing" mechanism in open-source applications.

The stealth of SSCAs is enabled by other factors. One of them is the exploitation of implicit trust and trusted mechanisms between different stakeholders within the same supply chain. By compromising a codebase before its compilation – or subverting certificates and signing protocols – an adversary can steadily decrease the likelihood that his intrusion is detected by anti-virus and monitoring tools. On top of that, threat actors can use obfuscation and evasion techniques to avoid detection by human operators and program analysis tools. One interesting example is Trojan source attacks, which aim to make malicious code appear different to the compiler than to the human eye.[18]

Similarly, end-users – who are often also the end targets – have limited visibility into the software creation and distribution process, thus reducing their ability to pre-

[12] Zack Whittaker, "Fake Android apps used for targeted surveillance found in Google Play," ZDNet, April 16, 2018, https://www.zdnet.com/article/fake-android-apps-used-for-targeted-surveillance-found-in-google-play/. For more information, see Table 5 on app store attacks.

[13] Trey Herr et al. "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[14] Oleg Demidov and Giacomo Persi Paoli, "Supply Chain Security in the Cyber Age – Sector Trends, Current Threats, and Multi-Stakeholder Responses," UNIDIR, February 2020, https://unidir.org/sites/default/files/2020-02/Technical%20Compendium%20for%20Supply%20Chain%20Security%20in%20the%20Cyber%20Age.pdf.

[15] Duan et al., "Towards Measuring Supply Chain Attacks on Package Managers for Interpreted Languages," ArXiv, December 2020, https://arxiv.org/abs/2002.01139.

[16] Vu et al., "Poster: Towards Using Source Code Repositories to Identify Software Supply Chain Attacks," In: CCS '20: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, October 2020, pp. 2093-2095, http://securitylab.disi.unitn.it/lib/exe/fetch.php?media=research_activities:experiments:ccs2020poster.pdf.

[17] Lily Hay Newman, "A Year After the SolarWinds Hack, Supply Chain Threats Still Loom," Wired, December 8, 2021, https://www.wired.com/story/solarwinds-hack-supply-chain-threats-improvements/.

[18] Nicholas Boucher and Ross Anderson "Trojan Source: Invisible Vulnerabilities," ArXiv, October 2021, https://arxiv.org/abs/2111.00169.

vent its compromise. This is especially true for equipment and services that are delivered with pre-installed software or software based on open-source code. This can lead to situations in which end-users are unaware about the software, library, or code that is integrated into the products they are using.[19] Therefore, even when a compromise in a product is known, it can be difficult to figure out what other software, libraries, or codebases are similarly affected, and how and where to check if they are compromised. This conundrum was recently illustrated by the Codecov incident, whereby a manufacturer of a software auditing tool informed its customers about a security breach in its Bash Uploader product. The corresponding script was used by thousands of its customers and was also integrated into various other programs, rendering it difficult to identify who and what was actually affected.[20]

2.4.2 Economic Espionage

Given the operational advantages of SSCAs, they are often leveraged to conduct economic espionage. Depending on the adversary's objectives, targets range from industrial to high-tech companies. Threat actors might be looking for strategic intellectual property or valuable business information. This might include access and theft of source code, which could allow further exploitation down the line. The Codevoc campaign is again a good example as multiple Codecov customers reported that the attacker was able to access their source code using the stolen information from the Codecov breach. Another good example of an economic espionage campaign was documented by the US Grand Jury indictments of two Chinese hackers in July 2020 who – while working for both the Chinese Ministry of State Security and their own financial gain – launched a global computer intrusion campaign that targeted intellectual property and confidential business information using, amongst other techniques, SSCAs.[21]

While generally considered to be the remit of APTs and cybercriminals, SSCAs and intellectual property theft have also been leveraged by hacktivists. Third-party breaches were the cause for the Panama and Paradise Paper leaks. In the case of the Panama papers, the 2016 hack of Panamanian law firm Mossack Fonseca was possible because of the company's outdated and flawed front-end security, which allowed the hacktivists to exfiltrate 2.6 terabytes of sensitive client data.[22] In the Paradise Paper case, around 1.5 terabytes of confidential offshore investment documents were accessed via a third-party law firm headquartered in Bermuda named Appleby.

2.4.3 Sabotage, Damage, or Disruption

SSCAs are also leveraged to conduct sabotage, disruption, and campaigns aimed at data destruction. A very prominent example is the 2017 NotPetya SSCA ransomware campaign which inserted a backdoor into the Ukrainian Me.Doc taxation software, which in turn crippled a wide array of critical infrastructure companies around the world, including logistics organizations, utilities, and banks. The inability of NotPetya to decrypt the encrypted files strongly suggests that the attack was primarily designed to cause destruction.[23] It is estimated that the NotPetya campaign caused around 10 billion USD in damages worldwide. As of this writing, it is still considered the costliest cyberattack to date.

A sector-specific example is the group known as Dragonfly 2.0. Dragonfly 2.0 targeted companies in the US and European energy sector in 2015. According to Symantec, they were using "a variety of infection vectors in an effort to gain access to a victim's network, including malicious emails, watering hole attacks, and Trojanized software."[24] Symantec went on to warn that "the Dragonfly group appears to be interested in both learning how energy facilities operate and also gaining access to operational systems themselves, to the extent that the group now potentially has the ability to sabotage or gain control of these systems should it decide to do so."[25]

An example of a non-state actor sabotage case which falls into the SSCA category is that of Marak Squires. Marak was the developer of two very popular open-source libraries known as 'colors' and 'faker.' In early January 2022, Marak intentionally decided to introduce

[19] Swiss Federal Intelligence Service, "Switzerland's Security 2021," https://www.newsd.admin.ch/newsd/message/attachments/67047.pdf.

[20] Ibid.

[21] US Department of Justice, "United States of America vs. Li Xiaoyu and Dong Jiazhi," Unites States District Court for the Eastern District of Washington, July 7, 2020, https://www.justice.gov/opa/press-release/file/1295981/download.

[22] James Temperton and Matt Burgess, "The security flaws at the heart of the Panama Papers," Wired, April 6, 2016, https://www.wired.co.uk/article/panama-papers-mossack-fonseca-website-security-problems.

[23] Andy Greenberg, "The Untold Story of NotPetya, the Most Devastating Cyberattack in History," Wired, August 22, 2018, https://www.wired.com/story/notpetya-cyberattack-ukraine-russia-code-crashed-the-world/.

[24] Symantec Threat Hunter Team, "Dragonfly: Western energy sector targeted by sophisticated attack group," Symantec, October 20, 2017, https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/dragonfly-energy-sector-cyber-attacks.

[25] Ibid.

a malign commit into colors and rolled out a new version for faker. Both changes resulted in an infinite loop that bricked thousands of projects that were dependent on these libraries. Marak essentially self-sabotaged as a mean to protest the free open-source project use by corporations and commercial entities that do not financially give back to the community.[26] In the context of the conflict in Ukraine, several other instances of self-sabotage or weaponization of open-source libraries were noted.[27]

2.4.4 Surveillance

Given that SSCAs can be leveraged to conduct espionage, they can also by extension be leveraged to conduct mass and targeted surveillance. While not as frequently discussed, SSCAs have in fact been leveraged to install spyware on numerous targets, including journalists, dissidents, and politicians. Many of these instances involve fake application that were advertised on legitimate app stores.[28] Surveillance campaigns can also take other forms. In Operation Nightscout for example, researchers discovered in January 2021 an SSCA that compromised the update mechanism of NoxPlayer, an Android emulator for PCs and Macs that is also part of BigNox's wide product range. Three different malware families were identified as being part of the campaign which distributed tailored malicious updates to selected victims based in Taiwan, Hong Kong, and Sri Lanka. The campaign had no known financial motives and was later linked to a group ESET calls Gelsemium.[29] According to ESET, Gelsemium operates in a very targeted manner in Asia and across the Middle East, and "considering its capabilities, this points to the conclusion that the group is involved in cyberespionage."[30]

2.4.5 Cybercrime

Lastly, SSCAs can be used to conduct a variety of cybercriminal activities. Among the most prominent and potent instances are the use of SSCAs to disseminate ransomware. The 2021 Kaseya incident is quite illustrative of this trend. It affected over 1,500 companies due to a zero-day vulnerability that allowed for remote code execution, which in turn facilitated the infection of the company's Virtual System/Server Administrator software (VSA) update with ransomware.[31]

SSCAs have also been used to conduct cryptocurrency heists. In 2021, three North Korean state-linked hackers (RGB/ Lazarus) were indicted by a US Grand Jury for cybercrimes which among others included the use of malicious cryptocurrency applications to target specific computers at designated cryptocurrency companies.[32] As the indictment explains, "The hackers would access the computer(s) of the victim cryptocurrency company without authorization and attempt to move through the victim cryptocurrency company's computer network in order to access a computer that would provide access to the victim cryptocurrency company's cryptocurrency wallet(s) and private keys to the wallet."[33]

SSCAs have also been leveraged to steal sensitive personal and financial data. In 2013 for example, hackers phished and subsequently compromising a third-party heating, ventilation, and air conditioning vendor that had access to the network of US retail giant Target. As a result, the hackers were able to breach Target to steal 40 million sets of credit and debit card data, and the personally identifiable information of 70 million Target customer accounts.[34]

[26] Ax Sharma, "npm Libraries 'colors' and 'faker' Sabotaged in Protest by their Maintainer—What to do Now?" Sonatype, January 10, 2022, https://blog.sonatype.com/npm-libraries-colors-and-faker-sabotaged-in-protest-by-their-maintainer-what-to-do-now.

[27] Lily Hay Newman, "The Fragile Open Source Ecosystem Isn't Ready for 'Protestware,'" Wired, March 25, 2022, https://www.wired.com/story/open-source-sabotage-protestware/.

[28] See Table 5 on app store attacks.

[29] ESET, "Gelsemium: When threat actors go gardening," We Live Security, June 9, 2021, https://www.welivesecurity.com/2021/06/09/gelsemium-when-threat-actors-go-gardening/.

[30] ESET, "ESET Research uncovers latest version of Gelsemium: Cyberespionage against government and other targets in Asia," June 9, 2021, https://www.eset.com/in/about/newsroom/press-releases/research/eset-research-uncovers-latest-version-of-gelsemium-cyberespionage-against-government-and-other-targ/; ESET, "Operation NightScout: Supply-chain attack targets online gaming in Asia," We Live Security, February 1, 2021, https://www.welivesecurity.com/2021/02/01/operation-nightscout-supply-chain-attack-online-gaming-asia/.

[31] Dan Goodin, "Up to 1,500 businesses infected in one of the worst ransomware attacks ever," ArsTechnica, July 6, 2021, https://arstechnica.com/gadgets/2021/07/up-to-1500-businesses-infected-in-one-of-the-worst-ransomware-attacks-ever/.

[32] US Department of Justice, "Three North Korean Military Hackers Indicted in Wide-Ranging Scheme to Commit Cyberattacks and Financial Crimes Across the Globe," February 17, 2021, https://www.justice.gov/opa/pr/three-north-korean-military-hackers-indicted-wide-ranging-scheme-commit-cyberattacks-and.

[33] US District Court for the Central District of California, "United States of America v. Jon Chang Hyok, Kim Il, and Park Jin Hyok," justice.gov, December 8, 2020, p. 15, https://www.justice.gov/opa/press-release/file/1367701/download.

[34] Maggie McGrath, "Target Data Breach Spilled Info On As Many as 70 Million Customers," Forbes, January 10, 2014, https://www.forbes.com/sites/maggiemcgrath/2014/01/10/target-data-breach-spilled-info-on-as-many-as-70-million-customers/?sh=636022b2e795.

2.5 Techniques, Tactics and Procedures

Operationally, SSCAs are usually trying to acquire unauthorized access to trusted ICT components, systems, services, and processes to insert themselves into the product development and life cycle to achieve an end goal. This can be espionage, disruption, surveillance, and/or financial crime. The complexity and variability of product lifecycles offer threat actors a wide attack surface to leverage a myriad of different SSCA techniques, tactics, and procedures (TTPs).

Threat vectors can range from unspectacular hacks, such as compromising a vendor's download site and replacing the links to legitimate patches with links to malicious ones,[35] to much more sophisticated hacks, such as exploiting one or multiple zero-day vulnerabilities to breach a product development server to change a product's source code before it is compiled, signed, or distributed.

The literature conceptualizes two overarching types of SSCA and TTP frameworks. The first type focuses on describing the lifecycle of an SSCA itself, while the second looks at the potential threat vectors in each phase of a software's lifecycle. The two types of frameworks are not mutually exclusive and often go together when presenting a comprehensive SSCA picture.

2.5.1 Attack Technique Catalogues

The first type of framework – including its associated taxonomies – seeks to assess the application, purpose, and customization of TTPs for conducting SSCAs. While other frameworks probably exist, four practical ones are widely discussed and used in the academic and cyber threat intelligence (CTI) and infosec literature.

2.5.1.1 Cyber Kill Chain

Developed by US defense giant Lockheed Martin, the cyber kill chain framework was designed to identify various steps an attacker must take to achieve its goals.[36] These generally encompass versions of the following phases:

1. Reconnaissance
2. Weaponization
3. Delivery
4. Exploitation
5. Installation
6. Command and control
7. Action on objectives

FireEye / Mandiant have produced versions of the Cyber Kill Chain that can also be used to analyze SCCAs.[37] For example, US cybersecurity company Virsec used the Cyber Kill Chain framework in 2020 to analyze the SolarWinds/Sunburst campaign.[38] Despite this, the Cyber Kill Chain framework has also received its fair share of criticism, notably by ENISA, who noted that the kill chain offers very generic classifications that do not allow for an in-depth analysis of supply chain attacks, and therefore makes comparing different types of SSCAs rather difficult.[39] Additionally, the Cyber Kill Chain framework does not differentiate between intermediate and end targets.

2.5.1.2 MITRE ATT&CK

The MITRE ATT&CK framework is a curated knowledge model that maps individual steps of threat actor behavior, techniques, and tactics within a campaign.[40] MITRE has recognized, analyzed, and catalogued supply chain attacks since at least 2013, when it published a report dubbed "Supply Chain Attack Framework and Attack Patterns," alongside a catalogue of 41 different types of attack patterns that could affect the US Department of Defense (DoD).[41] In 2017, MITRE published a follow-up study that delineated three broad conceptual methods of supply chain attacks:[42]

- Insertion: Adding information, code, or functionalities to an ICT module or component,

[35] US NIST-CSRC, "Software Supply chain Attacks", 2017, https://csrc.nist.gov/CSRC/media/Projects/Supply-Chain-Risk-Management/documents/ssca/2017-winter/NCSC_Placemat.pdf.

[36] Lockheed Martin, "The Cyber Kill Chain," https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html.

[37] FireEye, "M-Trends 2010: The Advanced Persistent Threat," FireEye.com, p. 3, https://www.christiandve.com/wp-content/uploads/2018/05/M-Trends.pdf.

[38] Satya Gupta, "Taxonomy of The Attack on SolarWinds and Its Supply Chain," December 23, 2020, https://www.virsec.com/hubfs/SolarWinds_Technical_Brief_2020.pdf?hsLang=en.

[39] European Union Agency for Cybersecurity, "ENISA Threat Landscape for Supply chain attacks," July 2021, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport.

[40] Mitre, "MITRE ATT&CK," https://attack.mitre.org/.

[41] John F. Miller, "Supply Chain Attack frameworks and Attack Patterns," MITRE, December 2013, https://www.mitre.org/sites/default/files/publications/supply-chain-attack-framework-14-0228.pdf; Kaspar Rosager Ludvigsen, Shishir Nagaraja, and Angela Daly, "Preventing or Mitigating Adversarial SupplY Chain Attacks; a legal analysis," August 9, 2022, https://arxiv.org/pdf/2208.03466.pdf.

[42] Heinbockel et al. "Supply Chain Attacks and Resiliency Mitigations," Mitre, 2017, https://www.mitre.org/sites/default/files/2021-11/pr-18-0854-supply-chain-cyber-resiliency-mitigations.pdf.

which performs a new, malicious function or otherwise subverts existing ones.
- Substitution: The complete replacement of an existing module or component in order to maliciously change its intended function or operation.
- Modification: Any change to the existing design or information that defines the system under development. In most cases, these changes will cause a degradation or introduce weaknesses during later stages of development or production.

The MITRE ATT&CK framework identifies "supply chain compromise" as a specific technique and even outlines several sub-techniques. Broadly speaking, it asserts that supply chain compromise can take place at any stage of the supply chain through:[43]

- Manipulation of development tools
- Manipulation of a development environment
- Manipulation of source code repositories (public or private)
- Manipulation of source code in open-source dependencies
- Manipulation of software update/distribution mechanisms
- Compromised/infected system images (multiple cases of removable media infected at the factory)
- Replacement of legitimate software with modified versions
- Sales of modified/counterfeit products to legitimate distributors
- Shipment interdiction

One critique of the MITRE ATT&CK framework has been that while it is particularly useful for companies to identify supply chain risks, it remains too generic and simplistic because it is solely focusing on the supply chain attacks themselves.[44]

2.5.1.3 CAPEC

Like MITRE ATT&CK, the Common Attack Pattern Enumeration and Classification (CAPEC) framework is a community resource developed and maintained by the MITRE Corporation with the support of the US Department of Homeland Security (DHS) for identifying and understanding attack patterns. It includes a dedicated supply chain category, which is classified as a "domain of attack" (ref. 437) under which an extensive list of attack patterns is listed. The overarching meta-attack patterns that extend beyond software aspects include:

- Excavation
- Configuration / Environment manipulation
- Software integrity attack
- Modification during manufacture
- Manipulation during distribution
- Hardware integrity attack
- Malicious logic insertion

Standard attack patterns include, for instance, malicious software updates (ref. 186), spoofing (ref. 657), and development alterations (ref. 444) through the insertion of malicious logics into a product software by an authorized developer (ref. 443) – meaning an insider attack.

2.5.1.4 ENISA

ENISA developed its own framework in 2021 to describe and analyze supply chain attacks. In comparison to ATT&CK and CAPEC, ENISA opted for placing specific emphasis on four key aspects of the supplier-customer relationship.[45] These are:

- Supplier: An entity that supplies a product or service to another entity.
- Supplier Assets: Valuable elements used by the supplier to produce the product or service.
- Customer: An entity that consumes the product or service produced by the supplier.
- Customer Assets: Valuable elements owned by the customer.

For each category, ENISA's framework describes the potential attack techniques that might be leveraged by a threat actor (see figure 2), whilst emphasizing that a supply chain attack is always a combination of at least two attacks: one against a supplier and one against the customer. In other words, one attack grants initial access to the upstream supplier before another attack gains access to the downstream consumer.

[43] Mitre, "MITRE ATT&CK – Supply Chain Compromise," https://attack.mitre.org/techniques/T1195/.

[44] European Union Agency for Cybersecurity, "ENISA Threat Landscape for Supply chain attacks," July 2021, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport.

[45] Ibid.

SUPPLIER		CUSTOMER	
Attack Techniques Used to Compromise the Supply Chain	Supplier Assets Targeted by the Supply Chain Attack	Attack Techniques Used to Compromise the Customer	Customer Assets Targeted by the Supply Chain Attack
Malware Infection	Pre-existing Software	Trusted Relationship [T1199]	Data
Social Engineering	Software Libraries	Drive-by Compromise [T1189]	Personal Data
Brute-Force Attack	Code	Phishing [T1566]	Intellectual Property
Exploiting Software Vulnerability	Configurations	Malware Infection	Software
Exploiting Configuration Vulnerability	Data	Physical Attack or Modification	Processes
Open-Source Intelligence (OSINT)	Processes	Counterfeiting	Bandwidth
	Hardware		Financial
	People		People
	Supplier		

Figure 2: ENISA's Supplier-Customer Supply Chain Framework[46] (ENISA, 2021)

2.5.1.5 Atlantic Council

In its study titled "Breaking Trust," the Atlantic Council, a US think tank, provides a list of data points which allow for the analysis of a supply chain attack.[47] The approach is somewhat like ENISA's as it focuses on:

- Date of initial access
- Attack/ disclosure
- Affected code
- Code owner/ location
- Downstream Target
- Affected codebase
- Attack vector
- Distribution vector
- Supply chain potential
- Impact

While useful, all the threat models and frameworks considered above present inherent limitations as they are anthropocentric by design and lack contextual application. As highlighted by Barabanov et al., these threat frameworks never consider non-human made threats, such as natural disasters or natural information leakages.[48] Moreover, they lack specificity, meaning that the lists of threats are never exhaustive and ought to always be defined and identified according to a specific ICT system or software development environment.

2.5.2 Attacks along the Software Lifecycle

The second type of frameworks pertaining to SSCAs focuses on the perimeter of threats/vulnerabilities (or threats/attack vectors) along a software supply chain. In other words, these frameworks put the emphasis on the different lifecycle stages of an ICT product or service and the different risks of compromise at each stage. As for any framework, there are a multitude of variations and visualizations of the same conceptual idea. For example, the US National Institute of Standards and Technology (NIST) report on *ICT Supply Chain Lifecycle*, the US Cybersecurity and Infrastructure Security Agency (CISA) publication on *Defending Against Software Supply Chain Attacks*, and the US's National Counterintelligence and Security Center (NCSC), discern between five to six different stages:[49]

1. Design
2. Development and production
3. Distribution
4. Acquisition and deployment
5. Maintenance (incl. improvements and updates)
6. Disposal or retirement

The Atlantic Council, in its 2020 study uses the following:

1. System design
2. Implementation
3. Iterative testing
4. Deployment
5. Update and maintenance

In its 2019 study called "ICT Supply Chain Integrity: Principles for Governmental and Corporate Policies," the Carnegie Endowment for International Peace uses a more detailed framework consisting of 11 phases.[50] NIST's key publication on Cybersecurity Supply Chain Risk Management Practices for Systems and Organizations (800-161) encompasses 10 phases.[51]

[46] Ibid., p. 7.

[47] Trey Herr et al. "Breaking Trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlantic-council.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[48] Barabanov et al. "On Systematics of the Information Security of Software Supply Chains," In: Radek Silhavy et al. "Software Engineering Perspectives in Intelligent Systems," Proceedings of the 4th Computational Methods in Systems and Software 2020, Vol. 1, p. 115-129, https://link.springer.com/chapter/10.1007/978-3-030-63322-6_9.

[49] NIST, "Defending Against Software Supply Chain Attacks," US Cybersecurity and Infrastructure Agency, April 2021, https://www.cisa.gov/sites/default/files/publications/defending_against_software_supply_chain_attacks_508_1.pdf.

[50] Ariel Levite, "ICT Supply Chain Integrity: Principles for Governmental and Corporate Policies," Carnegie Endowment for International Peace, October 4, 2019, https://carnegieendowment.org/2019/10/04/ict-supply-chain-integrity-principles-for-governmental-and-corporate-policies-pub-79974.

[51] Jon Boyens, Angela Smith, Nadya Bartol, Kris Winkler, Alex Holbrook, and Matthew Fallon, "Cybersecurity Supply Chain Risk Management Practices for Systems and Organizations," Draft NIST Special Publication 800-161 Revision 1, October 28, 2021, p. 243, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-161r1-draft2.pdf.

CAPEC, meanwhile, has a dedicated view for supply chain risks (ref. 683)[52] which is based on CISA's lifecycle framework.[53] It covers standards attack patterns across the following categories:

- Design (ref. 684)
- Development and Production (ref. 685)
- Distribution (ref. 686)
- Acquisition and Deployment (ref. 687)
- Sustainment (ref. 688)
- Disposal (ref. 689)

A major difficulty for cyber supply chain risk management is that many of these different phases are in the hands of different entities with various priorities, rationalities, resources, know-how, and security awareness. For an end target or customer, there is little control and oversight possible into all the things that occur upstream. In addition, due to the private and secretive nature of the development of proprietary code and even open-source code, there is little transparency or insight for downstream users as to the actual development process and the security steps taken. This lack of visibility creates a gap between visible, direct, and perceived supply chain risks and the invisible, indirect one's customers cannot protect themselves against. The general assumption is that the farther upstream the compromise occurs, the harder it is to detect and remediate for downstream users. Additionally, any upstream compromise allows for multiple paths to compromise downstream segments.

Based on the literature and existing models, the following paragraphs set out a broad overview of the different phases and potential attack vectors therein. The chosen phases are limited to five – like the framework advanced by the US NCSC – as most threats can be re-grouped into these. These five phases are:

1. Design
2. Development/ implementation
3. Distribution, acquisition, and deployment
4. Maintenance/ update
5. Disposal/ retirement

From an individual or product risk analysis perspective, it would make sense to delineate all the identifiable phases beyond these generic five.

The presented catalogue of threats is based on numerous source material, including technical bodies (MITRE, NIST, ITU), national cybersecurity organizations (US CISA, UK NCSC, FR ANSSI, NZ CERT, DE BSI, CH MELANI) and academic/policy entities (The Atlantic Council, Carnegie Endowment). Where possible, they are supplemented with case examples. An important caveat is that while this catalogue tries to cover a lot of ground, the standard for comprehensiveness continues to evolve as new techniques and sub-techniques are used and discovered.

2.5.2.1 Design

SSCAs can sometimes be thought of as roots planted directly into the design phase of a particular software – either willingly or unknowingly. For instance, a software designer that is coerced by a threat actor could intentionally insert malicious functions into a seemingly benign yet widely distributed software product. This could range from information stealing functions and cryptocurrency theft to disruptive outcomes.

An SSCA attack during this phase often entails an adversary that modifies the design of a product, technology, or component, to achieve some form of effect or malicious impact once the system is deployed.[54] As underlined by CAPEC, "design alteration attacks include modifying system designs to degrade system performance, cause unexpected states or errors, and general design changes that may lead to additional vulnerabilities. These attacks generally require insider access to modify design documents, but they may also be spoofed via web communications."[55]

While there is very little literature that considers supply chain attacks this early in a software's lifecycle, the US NCSC points out two known cases as to what those attacks might look like: The GoldenSpy campaign in 2020 and the AppleJeus campaigns in 2018.[56]

[52] CAPEC, "CAPEC View: Supply Chain Risks", September 2022, https://capec.mitre.org/data/definitions/683.html.

[53] CISA, "Supply Chain Risks for Information and Communication Technology". Cyber and Infrastructure Security Agency (CISA). 2018-12, https://www.cisa.gov/sites/default/files/publications/19_0424_cisa_nrmc_supply-chain-risks-for-information-and-communication-technology.pdf.

[54] CAPEC, "Design Alteration," October 21, 2021, https://capec.mitre.org/data/definitions/447.html.

[55] Ibid.

[56] Steve Zurier, "GoldenSpy' Malware Hidden in Tax Software Spies on Companies Doing Business in China," DarkReading, June 25, 2020, https://www.darkreading.com/threat-intelligence/-goldenspy-malware-hidden-in-tax-software-spies-on-companies-doing-business-in-china; Brian Hussey, "The Golden Tax Department and the Emergence of GoldenSpy Malware,"Trustwave, SpiderLabs Blog, June 25, 2020, https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/the-golden-tax-department-and-the-emergence-of-goldenspy-malware/; MITRE, "GoldenSpy," https://attack.mitre.org/software/S0493/.

In the GoldenSpy case it was discovered that a legitimate tax payment software mandated by the China Tax Bureau to conduct business in China had an embedded backdoor, which allowed for remote access and exfiltration of data to Chinese servers. The malware was first detected by US cybersecurity company Trustwave during a threat hunting operation on behalf of a UK-based technology company. While the identity of the UK company has not been publicly disclosed, the media outlet DarkReading explained that the company in question "has strong ties to the defense industry and does significant business in the US, Australia, and the UK."[57]

Due to the timing of most of the malware's attacks, which occurred in 2017, Trey Herr et al. put forward the notion that the campaign was likely influenced by then US President Trump's assertive China policy during his first year in office.[58] GoldenSpy also occurred amidst a longstanding Chinese government campaign of economic espionage and intellectual property theft targeting Western companies. Combined, these events have led some analysts to speculate that the malware was knowingly inserted by Chinese state-backed actors. As of this writing it is still unknown whether the Chinese company that developed and disseminated the tax software and/or the China Tax Bureau were aware of the backdoor.

In the AppleJeus campaign, North Korean APT Lazarus was able to successfully compromise several banks, fintech companies, and cryptocurrency exchanges through the use of a trojanized cryptocurrency trading application named Celas Trade Pro – which Lazarus recommended to company employees via social engineering.[59] Celas Trade Pro was in essence a modified version of the benign Q.T. Bitcoin Trader application. However, in contrast to Q.T. Bitcoin Trader, Celas Trade Pro's sole purpose was to deliver an update that included the malicious AppleJeus code immediately after the application was installed. Kaspersky explained that, "it looked like the threat actor had found an elaborate way to create a legitimate looking business and inject a malicious payload into a 'legitimate looking' software update mechanism."[60] Hence, instead of attacking or compromising a software supply chain, the threat actors just decided to create a fake business with a fake application. This was not the only known instance AppleJeus malware was disseminated via fake applications. Other versions of the AppleJeus malware were later found to be hidden in other fake applications promoted by legitimate-looking companies, such as UnionCrypto, JMT Trading, Kupay Wallet, CoinGo Trade, Dorusion, and Ant2Whale.[61]

AppleJeus is not the only campaign in which attackers have built their own malicious applications designed to appear legitimate.[62] Other examples include DroidDream in 2011 and Expensive Wall in 2017.[63] Researcher have also found that some adversaries simply repackage well-known and legitimate applications with malicious code before bundling it as a download item on third-party websites. Examples of such SSCAs include DroidJack RAT which was found in downloads for Pokémon Go, and the Geinimi Trojan which was discovered in repackaged download for Android games such as Monkey Jump 2 and City Defense.[64]

While the possibilities for downstream compromise in the design phase are truly endless, one could argue that this attack vector is more akin to the stealthy distribution of malware. As such, it is up for debate whether they constitute actual supply chain attacks. The lack of differentiation between supplier and customer, combined with the absence of a two-phased attack would not qualify them as SSCAs under ENISA's framework.

2.5.2.2 Development/ Implementation

The code development environment is subject to numerous trust boundaries and data flows. For instance, a high level of trust is implicitly required to exist between

[57] Steve Zurier, "GoldenSpy' Malware Hidden in Tax Software Spies on Companies Doing Business in China," DarkReading, June 25, 2020, https://www.darkreading.com/threat-intelligence/-goldenspy-malware-hidden-in-tax-software-spies-on-companies-doing-business-in-china.

[58] Trey Herr et al. "Breaking Trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[59] CISA, "Alert (AA21-048A) - AppleJeus: Analysis of North Korea's Cryptocurrency Malware," February 17, 2021, https://www.cisa.gov/uscert/ncas/alerts/aa21-048a; Kaspersky, "Operation AppleJeus: Lazarus hits cryptocurrency exchange with fake installer and macOS malware," Securelist, August 23, 2018, https://securelist.com/operation-applejeus/87553/; David Bisson, "Crypto Malware 'AppleJeus' Opens Cryptocurrency Wallets to Thieves," Security Intelligence, April 6, 2021, https://securityintelligence.com/news/applejeus-crypto-malware-targets-cryptocurrency/.

[60] Kaspersky, "Operation AppleJeus: Lazarus hits cryptocurrency exchange with fake installer and macOS malware," Securelist, August 23, 2018, https://securelist.com/operation-applejeus/87553/.

[61] CISA, "Alert (AA21-048A) - AppleJeus: Analysis of North Korea's Cryptocurrency Malware," February 17, 2021, https://www.cisa.gov/uscert/ncas/alerts/aa21-048a.

[62] Trey Herr et al., "App stores in focus," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/commentary/feature/app-stores-in-focus/.

[63] Check Point, "ExpensiveWall: A Dangerous 'Packed' Malware On Google Play That Will Hit Your Wallet," September 14, 2017, https://blog.checkpoint.com/2017/09/14/expensivewall-dangerous-packed-malware-google-play-will-hit-wallet/; Dennis Fisher, "Analysis Shows DroidDream Trojan Designed for Future Monetization," Threat Post, March 3, 2011, https://threatpost.com/analysis-shows-droiddream-trojan-designed-future-monetization-030311/74991/.

[64] Chris Brook, "Malicious Pokémon Go App Installs Backdoor on Android Devices," Threat Post, July 11, 2016, https://threatpost.com/malicious-pokemon-go-app-installs-backdoor-on-android-devices/119174/; Jack Clark, "Geinimi Trojan targets Android devices," ZDNet, December 21, 2010, https://www.zdnet.com/article/geinimi-trojan-targets-android-devices/.

the initial code writers, code submitters, contributors, and maintainers. That same trust also extents to the systems encompassing the codebase, the build system, and the distribution platform. This trust can easily be abused by threat actors. During the development of a software, system or service, there are numerous different threat vectors located upstream. Among these, the literature highlights three common ones:

1. Opensource code/software repositories (OSS) compromise
2. Software development kits (SDKs) compromise
3. Code-signing mechanism compromise

2.5.2.2.1 Compromising opensource code: Attacks against code libraries, repositories, packages, or dependencies

Interfering early in a software's lifecycle allows a threat actor to distribute malicious code downstream before any code-signing mechanisms are put in place. From an attacker's point of view, open or third-party code libraries and package repositories represent desirable, reliable, and scalable malware distribution channels, especially as many vulnerabilities can be hidden under a web of dependencies among these assets.

Open and third-party code libraries have become increasingly popular and unavoidable in everyday software development.[65] Indeed, modern software products are now assembled from ready-made coding components from a variety of suppliers: proprietary code, opensource components, and third-party APIs. No single developer can build a modern application on their own, and software reuse has become the norm. Even large-scale proprietary software like Windows and MacOS integrate large amounts of open-source code. The open-source community is one of the keystones of this development ecosystem and technological innovation. Without any strict or rigid organizational structure, its members rely on self-organization and collaboration to develop open-source software (OSS), which can either be project- or community-based.

Software developer (work) culture also contributes to the high degree of reliance and development of these OSS. For instance, in the West, coders are often encouraged to continuously and pro-actively contribute and commit to OSS. In general terms, their "ranks" or number of contributed projects is often regarded as a defining element of their competencies, seriousness, work ethic, and ability to give back to the community. This trend tends to generalize and normalize the use and development of OSS across organizations and projects. However, such an extensive use of OSS brings lots of issues or vulnerabilities. A first is the issue of oversight over code/coder origin and actual development practices. Many OSS projects accept contributions and modifications from loosely affiliated, effectively anonymous programmers whose names (or other metadata) can be spoofed. The public nature of such projects and the lack of transparency provided thus renders them even more potentially vulnerable to malicious code injection.

Another potential vulnerability is that while decentralized or crowdsourced auditing and discovery of vulnerabilities can be a good incentive for the use of OSS, few mechanisms exist to ensure efficiency or reliability. This becomes especially problematic with the exponential growth of OSS projects, which consequently has not only expanded the potential attack surface but also made auditing code more challenging. For example, the number of public repositories hosted on GitHub exploded from 46,000 in February 2009 to 28 million by January 2020.[66] This risk is reinforced by certain insecure practices. Recent research in developer security usability has documented that a significant portion of developers will copy and paste insecure source code without sufficient review from unofficial online sources such as Stack Overflow.[67] Moreover, code re-use often creates, among other things, dependencies with third-party libraries or packages. This creates additional vulnerabilities, as dependencies are often managed by dependency managers that automatically resolve, download, and install hundreds of open-source packages – with limited oversight.

Ruby (or Rubygems) – as an example of an opensource community codebase – is used by sites like Twitter, Hulu, and Shopify. Particularly the increasing popularity of program language-specific third-party package repositories – which are similar to Rubygems, such as NPM for Java Script and PyPI for Python – makes them attractive targets for SSCAs. SSCAs via open-source projects are not a new phenomenon. In fact, there is a rich history of

[65] For a holistic supply chain attack tree against Open-source software, please see Ladisa et al., "Taxonomy of Attacks on Open-Source Software Supply Chains, 2022, https://arxiv.org/abs/2204.04008.

[66] US NCSC, "Software Supply Chain Attacks," 2021, https://www.aha.org/system/files/media/file/2021/04/software-supply-chain-attacks-4-1-21.pptx.

[67] Yasemin Acar et al., "You get where you're looking for: The impact of information sources on code security," 2016 IEEE Symposium on Security and Privacy, https://usp.internet.byu.edu/static/papers/code-security-sp-2016.pdf; S. Fahl, "Stack overflow considered harmful? the impact of copy amp;paste on android application security," In: 2017 IEEE Symposium on Security and Privacy, pp. 121–136.

attempted insertion of backdoors into critical code bases. One historical example is the attempted insertion of a backdoor into the Unix kernel back in 2003.[68] Leveraging OSS for SSCAs has not only become more frequent, but also the attack vectors and techniques have become quite diverse.

One attack technique is stealing credentials, whereby an adversary steals the credentials of a package or library owner (or that of a trusted party) to subvert and poison the OSS components. Credentials can be stolen in various ways, including by spear-phishing or by deploying spyware against a trusted party. Similarly, the adversary might exploit flaws in access management systems, which then allows them to grant themselves the required privileges and access rights (for example, by making themselves admin).

Another attack technique is typosquatting, which is when an attacker publishes their own packages – with a built-in malicious payload – under a name that is like those of legitimate popular packages.[69] Due to the large number of package owners and the number of administrators, such attacks are likely to remain undetected for quite a long time. The PyPi NPM attack is a good example of typosquatting. The threat actor essentially split the package name into elements based on the "hyphen" character, and then rearranged the name "python-nmap" into "nmap-python." Another example of a typosquatting attack occurred in February 2020 against the Ruby repository. It involved two accounts that uploaded 700+ typosquatted packages, leading to more than 100,000 downloads of malware-ridden packages that were used to steal bitcoins.[70]

A variant of attacks with the same rationale includes package combosquatting.[71] Here the threat actors register fraudulent packages, not through a typo, but through a seemingly legitimate addition to the name. Vu, Plate and Sabetta compiled a list of combosquatting examples posing for PyPi (see figure 3).[72]

TABLE I: Malicious packages in our sample.

#	Time Appear	Malicious Package	Legitimate package	Names change	Levenshtein distance d d=1	d=2
1	2016-03-02	virtualnv	virtualenv	Delete 'e'	✓	
2	2016-03-03	mumpy	numpy	Substitute 'n' by 'm'	✓	
3	2017-05-01	crypt	crypto	Delete 'o'	✓	
4	2017-06-02	django-server	django-server-guardian-api	Delete "-guardian-api"		
5	2017-06-02	pwd	pwdhash.py	Delete "hash.py"		
6	2017-06-02	setuptool	setuptools	Delete 's'	✓	
7	2017-06-02	setup-tools	setuptools	Insert '-'	✓	
8	2017-06-02	telnet	telnetsrvlib	Delete "srvlib"		
9	2017-06-02	urlib3	urllib3	Delete 'l'	✓	
10	2017-06-02	urllib	urllib3	Delete '3'	✓	
11	2017-06-03	acquisition	acquisition	Delete 'i'	✓	
12	2017-06-03	apidev-coop	apidev-coop_cms	Delete "_cms"		
13	2017-06-04	bzip	bz2file	Substitute "2file" by "ip"		
14	2017-11-23	djanga	django	Substitute 'a' by 'o'	✓	
15	2017-11-24	easyinstall	easy_install	Delete '_'	✓	
16	2017-12-05	colourama	colorama	Delete 'u'	✓	
17	2018-04-25	opencv	opencv-python	Swap 'c' and 'v' & Delete "-python"		
18	2018-05-02	mateplotlib	matplotlib	Insert 'e'	✓	
19	2018-05-02	numipy	numpy	Insert 'i'	✓	
20	2018-05-02	python-mysql	MySQL-python	Swap "python" and "mysql"		✓
21	2018-05-03	libcurl	pycurl	Substitute "py" by "lib"		
22	2018-05-03	libhtml5	html5lib	Swap "html5" and "lib"		
23	2018-05-03	pysprak	pyspark	Swap 'a' and 'r'		✓
24	2018-05-03	PyYMAL	pyyaml	Swap 'a' and 'm'		✓
25	2018-05-10	nmap-python	python-nmap	Swap "nmap" and "python"		✓
26	2018-05-10	python-mongo	pymongodb	Delete "db" & Substitute "py" by "python-"		
27	2018-05-10	python-openssl	openssl-python	Swap "openssl" and "python"		✓
28	2018-09-17	pytz3-dev	pytz	Insert "3-dev"		
29	2018-10-29	python-sqlite	pysqlite	Substitute "py" by "python-"		
30	2018-10-30	python-ftp	pyftpdlib	Delete "dlib" & Substitute "py" by "python-"		
31	2018-10-30	python-mysqldb	MySQL-python	Swap "python" and "mysql" & Insert "db"		
32	2018-10-30	smb	pysmb	Delete "py"		✓
33	2018-10-31	pythonkafka	kafka-python	Swap "kafka" and "python" & Delete '-'		
34	2019-12-01	jellyfish	jellyfish	Substitute 'l' by 'l'	✓	
35	2019-12-01	python3-dateutil	python-dateutil	Insert '3'	✓	
36	2018-04-25	ssh-decorate	ssh-decorate	Hijacked Package		

Figure 3: List of recent combosquatting attacks against PyPi (Vu et al., 2019)

Name squatting is when an attacker is able to get their hands on the specific username (or a similar one) of a contributor, allowing them to impersonate established users. Previously, this was made possible because of the internal username policies of some platforms and/or public code libraries such as GitHub, which allowed one to claim inactive usernames. Some of these policies have changed in the past year.

Similar in spirit are repo jacking attacks (or dependency repository hijacking), whereby an attacker can hijack entire repositories (and its links) if the original owner has changed or deleted its username (or transfers its repository to another user before deleting his username). In October 2020, researchers at Security Innovation found a repo jacking campaign that exposed over 70,000 open-source projects, including popular projects and frameworks from Google, GitHub, and Facebook.[73]

Malicious commits refer to instances in which a malicious actor "commits" a seemingly benign alteration of code into a (public) repository. A good example is the 2021 attack on the official PHP Git repository, whereby an attacker pushed two malicious commits and disguised their alterations as a fix to a typo under the name of the creator of PHP.[74] The rogue code inserted a backdoor into all the websites that implemented the infected repository.

[68] Nicholas Boucher and Ross Anderson, "Trojan Source: Invisible Vulnerabilities," Arxiv, 2021, https://arxiv.org/pdf/2111.00169.pdf.

[69] Nikolai Tschacher, "Typosquatting programming language package managers," June 8, 2016, https://incolumitas.com/2016/06/08/typosquatting-package-managers/.

[70] Tim Anderson, "Typosquatting RubyGems laced with Bitcoin-nabbing malware have been downloaded thousands of times," The Register, April 21, 2020, https://www.theregister.com/2020/04/21/rubygems_bitcoin_malware/.

[71] Panagiotos Kintis et al. "Hiding in Plain Sight: A Longitudinal Study of Combosquatting Abuse," Arxviv, August 28, 2017, https://arxiv.org/abs/1708.08519.

[72] Duc-Ly Vu et al. "Typosquatting and Combosquatting Attacks on the Python Ecosystem," 2020 IEEE European Symposium on Security and Privacy Workshops, p. 511, https://ieeexplore.ieee.org/document/9229803/.

[73] Indiana Moreau, "Repo Jacking: Exploiting the Dependency Supply Chain," Security Innovation, October 22, 2020, https://blog.securityinnovation.com/repo-jacking-exploiting-the-dependency-supply-chain.

[74] Malwarebytes Labs, "Malicious commits found in PHP code repository: What you need to know," March 30, 2021, https://blog.malwarebytes.com/hacking-2/2021/03/malicious-commits-found-in-php-code-repository-what-you-need-to-know/; Ax Sharma, "PHP's Git server hacked to add backdoors to PHP source code," BleepingComputer, March 29, 2021, https://www.bleepingcomputer.com/news/security/phps-git-server-hacked-to-add-backdoors-to-php-source-code/.

Brandjacking describes the impersonation of a well-known package.[75] One known case was discovered when an attacker published a malware on the NPM registry called "web-browserify," in an attempt to imitate the legitimate "browserify" component.

Dependency[76] confusion (also known as repo name squatting, dependency hijacking, namespace confusion, or supply chain substitution attack) is when "a software installer script is tricked into pulling a malicious file from a public repository instead of the intended file of the same name from an internal repository."[77] Security researcher Alex Birsan – who also gave this attack vector its name – has successfully run and documented working dependency confusions even against companies like Apple, Uber, Tesla, Shopify and Microsoft.[78] His (ethical) attacks consisted of uploading malware to open-source repositories such as PyPI, NPM, and RubyGems, and naming his repos in such a way that they would be downloaded and used by the target company's application due to inherent flaws in their design.

Library masking is an attack technique that was highlighted by Barabanov et al., whereby an attacker exploits the fact that some internal and external components in a code library often have the same names and that code compilers/assemblers often give preference to a later version.[79] An attacker can thus insert a "recent" malicious external component into a library which will then in turn be compiled by the assembler due to its internal preferences.

Other attack paths are also known that use various methods to compromise existing packages or upload malicious code under the names of dependencies that no longer exist.[80] In collaboration with SAP Security, researchers at the University of Bonn built a dependency attack tree (see figure 4) that resulted out of their analysis of the code in 174 malicious components used in past supply chain attacks.[81]

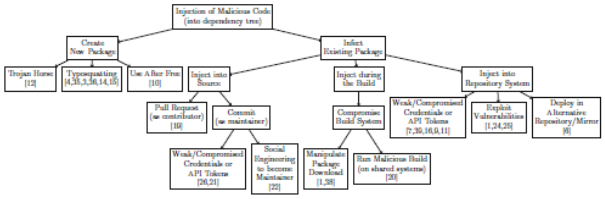

Figure 4: Attack tree to inject malicious code into dependency tree (Ohm & al., 2020).

Date Reported	Incident Name
2021	SushiSwap
2021	Birsan Research
2019	SneakerBots
2018	Octopus Scanner
2018	RubyGems Backdoor
2018	NPM Package hack
2018	Copay Compromise
2018	Event-stream Npm

Table 2: Various examples of recent (attempted) SSCA leveraging open-source software.

2.5.2.2.2 Compromising Software Development Tooling: Attacks against Programs, Tools, and Kits

Similar to the use of the aforementioned open-source libraries, packages, and dependencies, the application and software development ecosystem relies extensively on third-party software development tools, such as management and database tools, software design tools, configuration management tools, compilers, system build tools, and software performance testing and load testing tools. These can, in turn, also be exploited to conduct an SSCA.

A key vector for such attacks is software development kits (SDK), which are collections of software tools often found in one installable package that help the development of applications. Tailored for specific hardware platforms and operating systems, they can include compil-

[75] Ax Sharma, "Damaging Linux & Mac Malware Bundled Within Browserify npm Brandjack Attempt," Sonatype, April 13, 2021, https://blog.sonatype.com/damaging-linux-mac-malware-bundled-within-browserify-npm-brandjack-attempt.

[76] Dependencies are code modules packaged for easy consumption in application code that you write. They are mechanisms to enable code reusability for commonly solved problems and are imported into your applications.

[77] Alex Birsan, "Dependency Confusion: How I Hacked Into Apple, Microsoft and Dozens of Other Companies," Medium, February 9, 2021, https://medium.com/@alex.birsan/dependency-confusion-4a5d60fec610; Dhiyaneshwaran, "Dependency Confusion," Github, September 4, 2021, https://dhiyaneshgeek.github.io/web/security/2021/09/04/dependency-confusion/.

[78] Alex Birsan, "Dependency Confusion: How I Hacked Into Apple, Microsoft and Dozens of Other Companies," Medium, February 9, 2021, https://medium.com/@alex.birsan/dependency-confusion-4a5d60fec610.

[79] Barabanov et al., "On Systematics of the Information Security of Software Supply Chains," In: Radek Silhavy et al. "Software Engineering Perspectives in Intelligent Systems," Proceedings of the 4th Computational Methods in Systems and Software 2020, Vol. 1, https://link.springer.com/chapter/10.1007/978-3-030-63322-6_9.

[80] Alex Birsan, "Dependency Confusion: How I Hacked Into Apple, Microsoft and Dozens of Other Companies," Medium, February 9, 2021, https://medium.com/@alex.birsan/dependency-confusion-4a5d60fec610.

[81] Marc Ohm et al., "Backstabber's Knife Collection: A Review of Open Source Software Supply Chain Attacks," Arxiv, May 19, 2020, p. 7, https://arxiv.org/abs/2005.09535.

ers, debuggers, and software frameworks, but also libraries of reusable functions such as ads, analytics, and push notifications. For reference, the average Android mobile app implements 18.3 separate SDKs.[82] Some of the most used ones are AppsFlyer and Adjust, both of which provide app installation tracking and event recording functionality.[83]

Development tool attacks are when a threat actor finds a way to compromise these development programs that are innocuously included in the later software development, thus providing tremendous scale to their campaign. The fashion in which threat actors can compromise these SDK varies: many of the techniques previously discussed can be adapted. For instance, a similarly named malicious SDK could be published on a public or third-party file-sharing service. Alternatively, the provider of an SDK could be compromised through spearphishing and stolen credentials that will grant the attacker access to the SDK code. One could even contemplate a watering hole attack whereby a fake SDK download page is created to lure a potential developer into downloading and using a compromised version.

XcodeGhost, a malware detected in 2015, is a typical example of an SSCA targeting an SDK. To conduct this attack, a malicious actor was able to poison a version of Xcode (dubbed XcodeGhost), a development environment for iOS and OS X apps, before uploading it onto a Chinese file-sharing service – Baidu yunpan – outside Apple's control, preview, and review process. Any apps created with the compromised SDK leaked data to the adversary and allowed the malicious actor to phish for credentials and financial information. Some of these compromised applications were even accepted into Apple's App Store, including malicious versions of WeChat, WinZip, and China Unicom Mobile Office, eventually impacting over 500 users.[84]

The Twilio incident is another interesting example. Twilio is a cloud communication platform-as-a-service (CPaaS) company that powers communications for over 40,000 businesses and its APIs help developers add voice, video, messaging, and authentication capabilities to their apps.[85] Customers include Twitter, Netflix, Uber, Shopify, Morgan Stanley, Airbnb, Spotify, Yelp and eBay among others. In July 2020, Twilio disclosed that an attacker had injected malicious code into its SDK library through a misconfigured Amazon Web Service cloud object storage (S3) bucket.[86] The malicious code functioned as a traffic redirector and was known to be part of a long-running malicious advertisement campaign known as Hookads.

Date reported	Incident's name
2021	Apple Xcode
2020	Twilio
2019	SourMint
2019	Operation sheep
2019	Simbad
2015	XcodeGhost

Table 3: Various examples of recent SSCA leveraging SDKs

2.5.2.2.3 Undermining Software Integrity Protocols: Code Signing, Certificates, and Hashing

Integrity protocols are central to software development, design, and maintenance. This ecosystem relies upon numerous external sources and third-party code elements. Using code/hash-signing with public keys and certificates, they provide a trusted and cryptographically secure indicator which verifies that the software was approved by its developer and has subsequently not been altered.[87] In practice, the signature is generated by encrypting the hash of the code with a private key. The hash is then tested against a decrypted version using a public key. A match guarantees that the code received is the same as the one initially hashed and encrypted by the author. Traditionally, the software issuer is the only one with the private key.

By comparison, the certificate encompasses a set of information such as the name of the delivering authority, the software issuer, the creation and expiration date, and the public key. Its purpose is essentially to provide

[82] "A New SafeDK Data Trends Report Reveals the Current State of Android SDKs in the GDPR Era, Including Insights on Private Data That SDKs Attempt to Access, Market Leading Mobile SDKs and More," PR Newswire, July 30, 2018, https://www.prnewswire.com/news-releases/a-new-safedk-data-trends-report-reveals-the-current-state-of-android-sdks-in-the-gdpr-era-including-insights-on-private-data-that-sdks-attempt-to-access-market-leading-mobile-sdks-and-more-300688440.html.

[83] Statista, "Most popular installed attribution software development kits (SDKs) across Android apps worldwide as of November 2022," https://www.statista.com/statistics/1036027/leading-mobile-app-attribution-sdks-android/.

[84] Trey Herr et al., "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[85] Sergiu Gatlan, "Twilio exposes SDK, attackers inject it with malvertising code," BleepingComputer, July 22, 2020, https://www.bleepingcomputer.com/news/security/twilio-exposes-sdk-attackers-inject-it-with-malvertising-code/.

[86] Sergiu Gatlan, "Twilio exposes SDK, attackers inject it with malvertising code," BleepingComputer, July 22, 2020, https://www.bleepingcomputer.com/news/security/twilio-exposes-sdk-attackers-inject-it-with-malvertising-code/; Ionut Arghire, "Exposed Twilio SDK Abused for Malvertising Attack," SecurityWeek, July 23, 2020, https://www.securityweek.com/exposed-twilio-sdk-abused-malvertising-attack; Twilio, "Incident Report: TaskRouter JS SDK Security Incident - July 19, 2020," Twilio Blog, July 22, 2020, https://www.twilio.com/blog/incident-report-taskrouter-js-sdk-july-2020.

[87] US NCSC, "Software Supply Chain Attacks," 2021, https://www.aha.org/system/files/media/file/2021/04/software-supply-chain-attacks-4-1-21.pptx.

the consumer with the assurance that the code issuer has a private key linked to the public key.[88]
Software integrity protocols, such as code-signing and hashing mechanisms, are another attack vector that can be leveraged for an SSCA. While here considered in the context of the deployment phase, compromise can occur at any of the five stages in a software's lifecycle, including (1) the design phase, (2) the development phase by legitimatizing malicious code (3) the deployment phase by compromising file servers, and (4) in the maintenance phase by hijacking updates.

With these attack vectors, adversaries can self-sign their own malicious code thanks to either the original code being unsigned or by stealing the private keys and certificates from the original code author. This can occur in several ways, including by exploiting default passwords of poorly secured accounts, phishing, and cracking weak encryption algorithms. There also exists an entire criminal market for trading stolen certificates, whereby an adversary can buy what they need for their campaign. In addition, the certificates and private keys could also potentially be leaked online due to misconfiguration or carelessness on the part of the software company or the certifying authority. Private keys and access to certificates can also be attained by insiders or intermediate resellers. Meanwhile, the signature system might also be fundamentally broken or badly implemented, allowing an adversary to bypass this step altogether.

The literature is full of examples in which these software integrity protocols have been undermined or compromised by stolen and altered certificates. The prevalence of these two attack vectors is not surprising given that if the malicious code is not inserted pre-signature (i.e., in the development phase), attackers will find alternative ways to legitimize their code injection so it can be distributed without raising any red flags.

Based on its own dataset of over 117 supply chain attacks, Herr et al. put together the following chart (figure 5), which provides a good overview of the diversity of techniques used to bypass software integrity protocols.[89] It also shows the growth of these attack vectors and their increased relevance for conducting SSCAs.

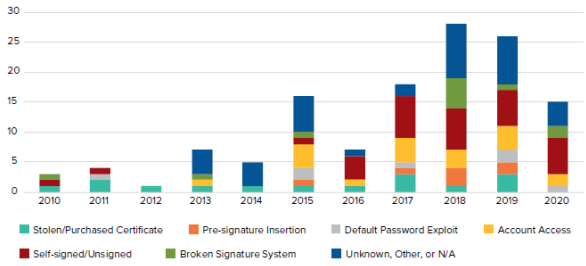

Figure 5: Repartition of attack vectors (Herr et al., 2020)[90]

Among the most discussed cases is the 2018 SSCA known as *Operation Shadowhammer.* In this campaign, a threat actor, dubbed Barium or Winnti, stole two legitimate ASUS certificates to push out backdoored updates to approximately one million machines. While the spillover was quite extensive, several researchers still believe that the end target was a very specific set of users and organizations.[91] The same group was possibly also behind a similar SSCA campaign in 2017 known as *ShadowPad*.

Date reported	Incident's name
2021	Teamviewer
2021	MimeCast
2019	NordVPN
2018	Shadowhammer
2017	Kingslayer
2017	CCleaner
2013	Java
2012	Duqu
2012	Adobe

Table 4: Various examples of SSCA leveraging compromised software integrity protocols

SSCAs conducted a decade ago have also prominently leveraged code-signing in their campaigns, including Duqu (2011), Adobe (2012), and Java (2013). The Duqu malware disguised itself as a driver file with a valid, albeit stolen, certificate in order to conduct reconnaissance on industrial systems. The threat actors hid their malware in the machine memory – installed via an exploit – before tricking the machine into loading those files from memory, and in the end stealing digital certificates from the affected machines. In the Adobe case, a malicious actor was able to compromise an internal

[88] Trey Herr et al., "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[89] Trey Herr et al., "Breaking trust: The dataset," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/resources/breaking-trust-the-dataset/.

[90] Trey Herr et al., "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, p. 15, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[91] Nicole Lindsey, "ASUS Supply Chain Attack Highlights New Security Vulnerability For Tech Giants," CPO Magazine, April 15, 2019, https://www.cpomagazine.com/cyber-security/asus-supply-chain-attack-highlights-new-security-vulnerability-for-tech-giants/; Kaspersky, "Operation ShadowHammer: a high-profile supply chain attack," Securelist, April 23, 2019, https://securelist.com/operation-shadowhammer-a-high-profile-supply-chain-attack/90380/.

Adobe build server which provided them access to the internal code-signing infrastructure, and the ability to create malware that was indistinguishable from legitimate Adobe software.[92] In the Java incident, an adversary leveraged stolen certificates to try to sign code for the purpose of installing a Trojan in Java versions 6 and 7.[93]

A more recent example includes the Mimecast incident in 2021 (see figure 6).[94] According to Mimecast's own incident report, APT29 – the same threat actor that conducted the SolarWinds campaign – used the SolarWinds compromise to gain access to part of Mimecast's production grid environment. As Mimecast explains, "[u]sing this entry point, the threat actor accessed certain Mimecast-issued certificates and related customer server connection information. The threat actor also accessed a subset of email addresses and other contact information, as well as encrypted and/or hashed and salted credentials. In addition, the threat actor accessed and downloaded a limited number of our source code repositories […]."[95] Mimecast did not find any evidence that APT29 had accessed emails or archival content the company held on the behalf of its customers.

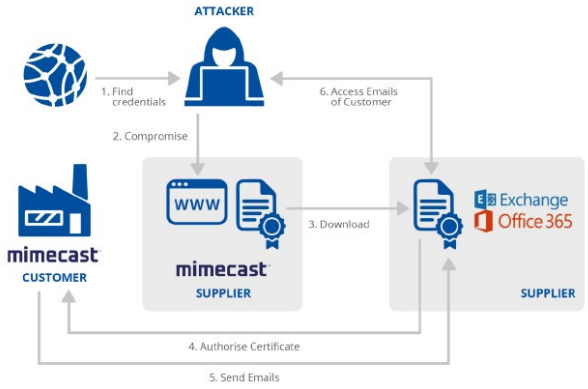

Figure 6: Mimecast Supply Chain attack (ENISA, 2021)

2.5.2.3 Distribution, Acquisition, Deployment

Software distribution, deployment systems, and processes can be subjected to various types of attacks. They might be independently targeted to inject and disseminate malicious code or can serve as unaffected distribution vectors for previously compromised code. Note: Malicious code can be disseminated either through the end target's own actions by for example downloading it or by a third-party through a software provider.

There are multiple ways to gain access to software distribution systems of third-party providers. For instance, a software provider could be infected by a credential-stealing spyware, granting the attacker direct access to the systems. Or an attacker could gain initial access through social-engineering ploys, which can range from phishing, fake apps, typo-squatting, to catfishing. Brute force attacks or watering-hole attacks can also grant access. As can the exploitation of specific software or configuration vulnerabilities in VPNs or other Internet-facing systems. Finally, some preliminary reconnaissance against the software provider itself could also allow an attacker to discover stolen and leaked credentials.

2.5.2.3.1 Managed Service Providers and Cloud Service Provider

Due to simplification of scalability and lower implementation cost, it is increasingly commonplace for organizations to outsource the management of their IT infrastructure to managed service providers (MSPs) and cloud service providers (CSPs).[96] The pandemic-induced increase in digitalization pressures combined with the shift toward remote working has similarly favored the demand for such services. According to a survey conducted by Altaro Software in late 2020, more than 75% of MSPs said that remote working was the best revenue-generating opportunity.[97] In similar vein, Kaseya's 2021 MSP Benchmark survey found that 65% of MSPs had increased their revenue by delivering cybersecurity services amidst the global economic downturn.[98]

[92] Kim Zetter, "Hackers Breached Adobe Server in Order to Sign Their Malware," Wired, September 27, 2012, https://www.wired.com/2012/09/adobe-digital-cert-hacked/.

[93] Trey Herr et al., "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[94] Mimecast, "Incident Report," https://www.mimecast.com/incident-report/; Tara Seals "Mimecast Certificate Hacked in Microsoft Email Supply-Chain Attack," Threat Post, January 21, 2021, https://threatpost.com/mimecast-certificate-microsoft-supply-chain-attack/162965/.

[95] Mimecast, "Incident Report," https://www.mimecast.com/incident-report/.

[96] SingCERT, "The Multiplier Effect – Targeting The MSP Supply Chain," August 5, 2021, https://www.csa.gov.sg/singcert/Publications/the-multiplier-effect---targeting-the-msp-supply-chain.

[97] Altaro, "80% of MSPs retain staff despite decreasing revenues due to COVID-19, Altaro survey shows," September 14, 2020, https://www.altaro.com/news/single/News-MSPs-retain-staff-despite-decreasing-revenues.php.

[98] Ester Shein, "Remote work and increased cybersecurity threats presented both challenges and opportunities for MSPs," TechRepublic, March 29, 2021, https://www.techrepublic.com/article/remote-work-and-increased-cybersecurity-threats-presented-both-challenges-and-opportunities-for-msps/.

MSPs and CSPs support overall IT management and administration tasks including network, application, infrastructure, and IT security. To do so, they are granted widespread access and administration rights to their client's IT systems. While useful and cost effective, this practice can be a double-edged sword. Concentrated privileged access to numerous organizations naturally attracts adversaries and may drastically increase the likelihood that a contracting organization will be impacted if their MSP or CSP provider is breached. This can be especially problematic because "these delegated administrative privileges are often neither audited for approved use nor disabled by a service provider or downstream customer once use has ended, leaving them active until removed by the administrators."[99]

There are several known SSCA cases that have targeted MSPs and CSPs. The most noteworthy ones encompass Kaseya (2021), Nobelium's targeting of MSPs and CSPs (2021), Magecart/Ticketmaster (2018), and APT10's Operation Cloud Hopper (2017). The rising threat of SSCA via MSP has been recognized by the cybersecurity authorities of the United Kingdom (NCSC-UK), Australia (ACSC), Canada (CCCS), New Zealand (NCSC-NZ), and the United States (CISA, NSA, FBI) in a joint call.[100]

Kaseya is a software service provider specializing in remote monitoring and management tools (RMM). One of its software products is called the Virtual System/Server Administrator (VSA), which it licenses out to its MSP clients. The MSP clients in turn provide various IT services to their own clients. In July 2021, the REvil ransomware group exploited a zero-day vulnerability in Kaseya's systems that allowed them to remotely execute commands on the VSA appliances of Kaseya's MSP clients in order to distribute a malicious update. As a result, the malicious update deployed ransomware on approximately 1,500 downstream MSP customers.

Nobelium – the threat actor accused of being behind the SolarWinds/Sunburst campaign – remained active throughout 2021 and pursued a variety of other targets. Among others, the group targeted several different resellers and technology service providers that customize, deploy, and manage cloud services and other technologies on behalf of their customers.[101] More precisely, Microsoft reported that Nobelium targeted around 140 MSPs and CSPs, with at least 14 successfully breached through a mix of password-spraying, brute-forcing, and phishing campaigns. Nobelium's secondary attack vector involved chaining together artifacts and access from across four distinct MSP/CSP providers to reach their end target (figure 7).

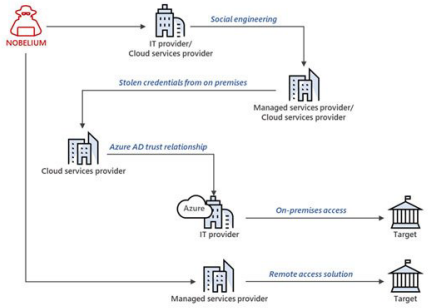

Figure 7: Nobelium's supply chain attack paths against MSPs (Microsoft, 2021)[102]

Operation Cloud Hopper was conducted by Chinese APT 10.[103] In April 2017, it was discovered that APT10 had executed a campaign against multiple MSPs across the globe to get their hands on their end target's corporate assets and trade secrets. The affected MSPs were compromised through over 70 different remote access Trojans and backdoors variants delivered through spearphishing (figure 8).

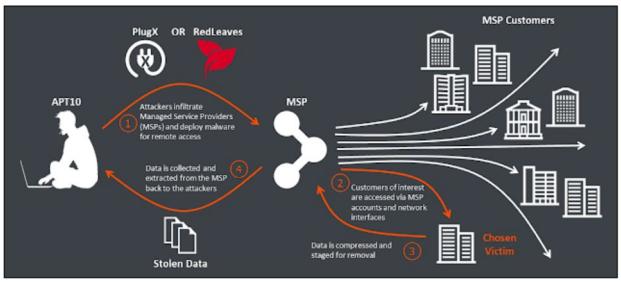

Figure 8: Cloud Hopper's attack stages (BAE system, 2017)[104]

[99] "NOBELIUM targeting delegated administrative privileges to facilitate broader attacks," Microsoft Threat Intelligence Center, October 25, 2021, https://www.microsoft.com/security/blog/2021/10/25/nobelium-targeting-delegated-administrative-privileges-to-facilitate-broader-attacks/.

[100] CISA, "Alert (AA22-131A): Protecting Against Cyber Threats to Managed Service Providers and their Customers," May 11, 2022, https://www.cisa.gov/uscert/ncas/alerts/aa22-131a.

[101] Ibid.

[102] "NOBELIUM targeting delegated administrative privileges to facilitate broader attacks," Microsoft Threat Intelligence Center, October 25, 2021, https://www.microsoft.com/security/blog/2021/10/25/nobelium-targeting-delegated-administrative-privileges-to-facilitate-broader-attacks/.

[103] TrendMicro, "Operation Cloud Hopper: What You Need to Know," April 10, 2017, https://www.trendmicro.com/vinfo/pl/security/news/cyber-attacks/operation-cloud-hopper-what-you-need-to-know; Adrian Nish and Tom Rowles, "APT10 – Operation Cloud Hopper," BAE Systems Threat Research Blog, April 3, 2017, https://baesystemsai.blogspot.com/2017/04/apt10-operation-cloud-hopper_3.html; BAE & PwC, "Operation Cloud Hopper," April 2017, https://www.pwc.co.uk/cyber-security/pdf/pwc-uk-operation-cloud-hopper-report-april-2017.pdf.

[104] Adrian Nish and Tom Rowles, "APT10 – Operation Cloud Hopper," BAE Systems Threat Research Blog, April 3, 2017, https://baesystemsai.blogspot.com/2017/04/apt10-operation-cloud-hopper_3.html.

2.5.2.3.2 Content Delivery Networks

Content delivery networks, or CDNs, are regionally dispersed proxy servers and data centers. Their purpose is to provide continuous availability and high performance by spatially distributing the service relative to end user. Through caching data, CDNs can provide faster access to various web and streaming content.[105] Their services are utilized by both users and content owners/networks. As an essential part of the Internet ecosystem, CDN providers and software-as-a-service (SaaS) companies are an extremely attractive target for SSCAs. A key factor is the trusted relationship CDNs have with other organizations, as well as their capacity to enable users to upload and share content with a wide array of secondary targets.[106] Furthermore, targeting CDNs also allows for a malicious campaign to be deployed within a certain geographic area.[107]

There are a few known cases in which supply chain attacks have been leveraged against CDNs. One example is the DownAndExec campaign in Brazil in 2017.[108] The threat actor essentially hid a banking Trojan within a JavaScript snippet that was hosted on the infrastructure of a CDN provider. Interestingly enough, the snippet had some functionality to check geographic locations. When it was accessed from Brazil, it downloaded and activated the malware. Such a technique apparently provided high bandwidth for payload delivery and command and control operations. Another example pertains to a malware that affected Google's CDN back in 2018.[109] The malicious code was hidden inside the metadata fields of Pac-Man images hosted on Google's official CDN (googleusercontent.com). These images were deployed against certain social networks, such as Blogger.com and the discontinued Google+ social network. The malware would activate once downloaded by either a website or a user request. The apparent intent was to steal PayPal security tokens with the goal of bypassing the PayPal authentication process.

The last example concerns a CDN vulnerability that was patched in time but would have provided malicious actors with a potent backdoor to conduct widespread downstream campaigns. In July 2021, Cloudflare reported that it had fixed a vulnerability in its content delivery network for open-source libraries (cdnjs).[110] Used by nearly 13% of all websites, Cloudflare's open-source library serves more than 4,000 JavaScript and CSS libraries, making it the second-most popular CDN for JavaScript – behind Google Hosted Libraries.[111] The vulnerability would have allowed an attacker to "publish[ing] packages to Cloudflare's CDNJS using GitHub and npm, to trigger a Path Traversal vulnerability, and eventually remote code execution."[112]

2.5.2.3.3 Web-Based: Download Site Attacks

Download site attacks - or patch site attacks - are more easily identified as SSCAs. Examples include the Dragonfly campaign in 2013, Monpass in 2021, and Wizvera VeraPort in 2020.[113]

The Dragonfly campaign targeted industrial control system (ICS) software.[114] The malicious actor was able to compromise the websites of three different ICS software suppliers by infecting legitimate download files with a Trojan that was equipped with remote access functionalities.

---

[105] Akamai, "What is a CDN (Content Delivery Network)?" https://www.akamai.com/our-thinking/cdn/what-is-a-cdn.

[106] ProcessBolt, "Supply Chain Attacks and the Vulnerability of CDNs," n.d., https://processbolt.com/supply-chain-attacks.

[107] Ibid.

[108] Cassius Puodzius, "DownAndExec: Banking malware utilizes CDNs in Brazil," WeLiveSecurity, September 13, 2017, https://www.welivesecurity.com/2017/09/13/downandexec-banking-malware-cdns-brazil/.

[109] Catalin Cimpanu, "Google User Content CDN Used for Malware Hosting," BleepingComputer, July 20, 2018, https://www.bleepingcomputer.com/news/security/google-user-content-cdn-used-for-malware-hosting/; Denis Sinegubko, "Hiding Malware Inside Images on GoogleUserContent," Sucuri, July 18, 2018, https://blog.sucuri.net/2018/07/hiding-malware-inside-images-on-googleusercontent.html.

[110] Jonathan Ganz et al., "Cloudflare's Handling of an RCE Vulnerability in cdnjs," Cloudflare Blog, July 24, 2021, https://blog.cloudflare.com/cloudflares-handling-of-an-rce-vulnerability-in-cdnjs/; Ravie Lakshmanan, "CloudFlare CDNJS Bug Could Have Led to Widespread Supply-Chain Attacks," The Hacker News, July 17, 2021, https://thehackernews.com/2021/07/cloudflare-cdnjs-bug-could-have-led-to.html; Ax Sharma, "Critical Cloudflare CDN flaw allowed compromise of 12% of all sites," BleepingComputer, July 16, 2021, https://www.bleepingcomputer.com/news/security/critical-cloudflare-cdn-flaw-allowed-compromise-of-12-percent-of-all-sites/.

[111] W3Techs, "Usage statistics and market share of CDNJS for websites," https://w3techs.com/technologies/details/cd-cdnjs.

[112] Ax Sharma, "Critical Cloudflare CDN flaw allowed compromise of 12% of all sites," BleepingComputer, July 16, 2021, https://www.bleepingcomputer.com/news/security/critical-cloudflare-cdn-flaw-allowed-compromise-of-12-percent-of-all-sites/.

[113] Luigino Camastra et al. "Backdoored Client from Mongolian CA MonPass," Avast, July 1, 2021, https://decoded.avast.io/luigicamastra/backdoored-client-from-mongolian-ca-monpass/; Ravie Lkashmanan, "Mongolian Certificate Authority Hacked to Distribute Backdoored CA Software," The Hacker News, July 2, 2021, https://thehackernews.com/2021/07/mongolian-certificate-authority-hacked.html.

[114] A. L. Johnson, "Dragonfly: Western Energy Companies Under Sabotage Threat," Broadcom Endpoint Protection, June 30, 2014, https://community.broadcom.com/symantecenterprise/communities/community-home/librarydocuments/viewdocument?DocumentKey=7382dce7-0260-4782-84cc-890971ed3f17&CommunityKey=1ecf5f55-9545-44d6-b0f4-4e4a7f5f5e68&tab=librarydocuments; Nell Nelson, "Global Information Assurance Certification Paper," GIAC, January 18, 2016, https://www.giac.org/paper/gicsp/724/impact-dragonfly-malware-industrial-control-systems/148912.

Monpass is a major certification authority in Mongolia. Its website was compromised in 2021 with one of its client installers being trojanized and backdoored. The infected client was available to download for approximately one month, resulting in the infection of at least one known customer.

Wizvera is a South Korean company that specializes in ID verification, password management, and cloud certificates. [115] It also provides an integration installation program managing security software called VeraPort, which is mandatory to access governmental services and banking websites. In November 2020, VeraPort was targeted by the North Korean APT Lazarus. Lazarus was able to compromise a VeraPort-supported web server, which allowed the threat actor to conduct a configuration request, which then introduced a signed malicious binary. The inherent flaw in Wizvera's setup was that its integrity protocols only checked whether the binary had a valid digital signature but did not checker whether the signature was legitimate. [116]

2.5.2.3.4 App Stores

App stores, such as the Google Play Store, Apple's App Store, and Tencent's Myapp, have become popular vectors for SSCAs. These stores are central pillars of the app ecosystem as they are integral to the application development cycle, security reviews, and product dissemination to the public. By acting as a trusted and consolidated marketplace for third-party applications and their updates, these stores are of great appeal to the general public, who can easily search, download, and rate new apps. The resulting high and frequent download traffic can be exploited by threat actors.

Over the past few years, several concerns have been raised regarding the app store model and its vulnerabilities and weaknesses. Notable shortcomings include stores' inability to consistently detect obfuscated malware. [117] As previously illustrated, in the design and development phases, app store attacks in the form of downloading malicious application can take a variety of forms. Malicious actors can design their own apps to try to impersonate legitimate apps and their updates, repackage applications with malicious code, or compromise SDKs. Table 5 provides additional instances of app store campaigns.

Date Reported	Targeted App/ Platform
2021	Andr & iPh
2020	Joker malware family
2020	Chrome Web Store spyware
2020	Camero, CryptManager, CallCam
2019	Soraka/Sogo App Attack
2019	Sandworm Android Attack
2019	Apple's App Store 17
2019	Targeting of Egyptian Human Rights Activists
2019	Radio Balouch / AhMyth
2019	Operation Sheep
2019	SimBad
2019	Malicious Chrome extensions
2019	Golduck

Table 5: Recent examples of SSCAs leveraging app stores

2.5.2.3.5 Pre-installed malware

Some supply chain attacks do not even require any download from the user and have been found to completely bypass the app stores by pre-installing apps directly onto smartphones. This was notably the case in the Android-Triada incidents (2016-2019), where it was discovered that Chinese malicious actors had successfully placed a Trojan onto new Android devices as pre-packaged firmware by injecting it into the system library. [118] The malware was quite sophisticated. It hid in the machine's RAM and ran each time an app made a system log to infect a core OS process. All in all, 42 smartphone models where affected, allowing the malware to exfiltrate user data and redirect financial transactions. According to Lukasz Siewierski, a reverse engineer on Google's Android Security team, the assumption is that a third-party "vendor using the name Yehuo or Blazefire infected the returned system image with Triada" while adding new features. It is unclear whether this was a deliberate infection or simple carelessness on the vendor's end. [119]

[115] Anton Cherepanov & Peter Kálnai, "Lazarus supply-chain attack in South Korea," WeLiveSecurity, November 16, 2020, https://www.welivesecurity.com/2020/11/16/lazarus-supply-chain-attack-south-korea/.

[116] Ibid.

[117] Avi Bashan, "Mobile Security: Why App Stores Don't Keep Users Safe," DarkReading, March 24, 2016, https://www.darkreading.com/vulnerabilities-threats/mobile-security-why-app-stores-don-t-keep-users-safe.

[118] Lukasz Siewierski, "PHA Family Highlights: Triada," Google Security Blog, June 6, 2019, https://security.googleblog.com/2019/06/pha-family-highlights-triada.html; Brian Krebs, "Tracing the Supply Chain Attack on Android," KrebsonSecurity, June 25, 2019, https://krebsonsecurity.com/2019/06/tracing-the-supply-chain-attack-on-android-2/; Dan Goodin, "Google confirms that advanced backdoor came preinstalled on Android devices," ArsTechnica, June 6, 2019, https://arstechnica.com/information-technology/2019/06/google-confirms-2017-supply-chain-attack-that-sneaked-backdoor-on-android-devices/; John Snow, "Triada: organized crime on Android," Kaspersky, March 3, 2016, https://www.kaspersky.com/blog/triada-trojan/11481/; Dr. Web, "Dr.Web: Trojan preinstalled on Android devices infects applications' processes and downloads malicious modules," drweb.com, July 27, 2017, https://news.drweb.com/show/?i=11390&lng=en.

[119] Lukasz Siewierski, "PHA Family Highlights: Triada," Google Security Blog, June 6, 2019, https://security.googleblog.com/2019/06/pha-family-highlights-triada.html.

A similar case occurred at Unimax in 2020. Unimax supplies low-cost mobile phones, notably through the lifeline assistance program, which is run by the US Federal Communications Commission to make communications services affordable for low-income consumers.[120] In 2020, researchers at Malwarebytes Labs discovered that several of Unimax's mobile phones came with pre-installed spyware without any publicly shared indications as to how the supply chain was infected.[121]

2.5.2.4 Maintenance/Update

As briefly exemplified in the discussion of the role of app stores, software and applications often require maintenance and revisions in the form of patches, fixes, and updates. While regular updating and patching is a cybersecurity good practice, it is also important to note that updates and patches might introduce unforeseen flaws and can open the door for malicious exploitation. Updates generally entail some kind of action or agreement between the publisher and the user/customer. More often than not, this agreement is automatized and not regularly scrutinized because of the use of trusted mechanisms, such as code signing.

In the wild, hijacking updates has become a quite common attack vector. Malicious updates can for instance be used to insert backdoors into third-party software, applications, or open-source dependencies.[122] One example of such a backdoor is Shadowpad.[123] In 2017, researchers at Kaspersky Lab discovered the Shadowpad backdoor in NetSarang's server management software, which is used by hundreds of major businesses across the globe. When activated, Shadowpad allowed the attacker to download further malicious modules onto a customer's network or steal data in the process. According to Kaspersky Lab, the Shadowpad backdoor was likely introduced by hiding inside an infected software update.[124] When Kaspersky reported its findings to NetSarang – a prominent South Korean software company – an updated version of the software without the malicious code was quickly released. How the update was compromised is still unknown. It is also unclear how many instances of Shadowpad remain dormant on systems worldwide that have failed to install the non-malicious update. According to Sentinel Labs, Shadowpad is a privately sold modular malware platform whose clients include APT41 (Winnti).[125] Shadowpad has also been used in the SSCA against ASUS and the campaign against CCleaner.[126]

Another example involving the update function of open-source software is the 2014 GOM Player incident. GOM Player is a free South Korean media player that is very prominent across parts of Asia. In January 2014, the installation of a malicious GOM Player update compromised a machine in the control room at the nuclear reactor facility in Monju, Japan.[127] The malicious update introduced a version of Gh0st RAT, which has been linked to Chinese APT threat actors but whose source code is publicly available online. According to Accenture Security, more than 40,000 emails and training documents were stolen from the compromised machine.[128]

Other more high-profile campaigns have also included breaches via software updates. These includes Flame in 2010/2012, NotPetya in 2017, and CCleaner in 2018. To disseminate the Flame malware, the threat actor – most likely Unit 8200 of the Israeli Defense Forces – used a cryptographic attack that allowed it to forge Microsoft

[120] US Federal Communications Commission, "Lifeline Support for Affordable Communications," fcc.gov, January 5, 2023, https://www.fcc.gov/lifeline-consumers.

[121] Nathan Collier, "We found yet another phone with pre-installed malware via the Lifeline Assistance program," Malwarebytes Labs, July 8, 2020, https://blog.malwarebytes.com/android/2020/07/we-found-yet-another-phone-with-pre-installed-malware-via-the-lifeline-assistance-program/; Nathan Collier, "United States government-funded phones come pre-installed with unremovable malware," Malwarebytes Labs, January 9, 2020, https://blog.malwarebytes.com/android/2020/01/united-states-government-funded-phones-come-pre-installed-with-unremovable-malware/.

[122] Trey Herr et al. "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[123] Dan Goodin, "Powerful backdoor found in software used by >100 banks and energy cos," ArsTechnica, August 15, 2017, https://arstechnica.com/information-technology/2017/08/powerful-backdoor-found-in-software-used-by-100-banks-and-energy-cos/; Kaspersky, "ShadowPad: How Attackers hide Backdoor in Software used by Hundreds of Large Companies around the World," August 15, 2017, https://www.kaspersky.com/about/press-releases/2017_shadowpad-how-attackers-hide-backdoor-in-software-used-by-hundreds-of-large-companies-around-the-world.

[124] Kaspersky, "ShadowPad: How Attackers hide Backdoor in Software used by Hundreds of Large Companies around the World," August 15, 2017, https://www.kaspersky.com/about/press-releases/2017_shadowpad-how-attackers-hide-backdoor-in-software-used-by-hundreds-of-large-companies-around-the-world.

[125] Yi-Jhen Hsieh & Joey Chen, "ShadowPad | A Masterpiece of Privately Sold Malware in Chinese Espionage," Sentinel Labs, August 19, 2021, https://www.sentinelone.com/labs/shadowpad-a-masterpiece-of-privately-sold-malware-in-chinese-espionage/

[126] Ibid.

[127] Pierluigi Paganini, "IT administrator at Monju Nuclear Power Plant discovered that a malware-based attack infected a system in the reactor control room," Security Affairs, January 10, 2014, https://securityaffairs.co/wordpress/21109/malware/malware-based-attack-hit-japanese-monju-nuclear-power-plant.html; Mark Graham, "Context Threat Intelligence - The Monju Incident," Accenture Security, Context, February 19, 2014, https://web.archive.org/web/20220120234401/https://www.contextis.com/us/blog/context-threat-intelligence-the-monju-incident; Trey Herr et al. "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[128] Graham, "Context Threat Intelligence - The Monju Incident," Accenture Security, Context, February 19, 2014, https://web.archive.org/web/20220120234401/https://www.contextis.com/us/blog/context-threat-intelligence-the-monju-incident.

security certificates. The adversary then used these certificates to hijack the Windows Update mechanism via a network adversary-in-the-middle attack. Flame targeted a very limited number of systems across the Middle East – specifically Iran – to steal information by taking screenshots and recording conversations via a system's plugged-in microphone.[129] In the NotPeyta campaign, the Russian APT known as Sandworm compromised an update of the Ukrainian M.E.Doc tax software to deploy a wormable ransomware.[130] The ransomware used its worming capability in conjunction with Mimikatz to steal credentials from a system's memory to rapidly and efficiently infect network after network and client after client. CCleaner is a popular maintenance tool for Microsoft systems that was used as a vehicle in one of the most large-scale supply chain attacks to date.[131] First, the threat actor gained access to a developer account at Piriform – which is the British software company that develops CCleaner. From there, the adversary stole a valid CCleaner certificate and poisoned the CCleaner update with a multistage payload – the so-called Floxif backdoor. Owing to the popularity of CCleaner, the attacker was able to infect approximately 2.2 million customers worldwide. This SSCA also targeted 18 specific companies in an espionage effort to specifically gain access to microelectronic vendors including ASUS, Fujitsu, Samsung, Sony and others.[132] Table 6 provides additional examples of SSCAs leveraging hijacked updates and legitimate update mechanisms.

Date	Incident's name
2019	Second Winnti Group Gaming Attack
2019	ShadowHammer
2019	strong_password attack
2019	Abiss (CCleaner v.2)
2019	Sandworm Android Attack
2020	Able Desktop
2020	SolarWinds
2021	Clickstudios Passwordstate
2021	BigNox
2021	Kaseya
2021	Lavabird

Table 6: Examples of supply chain attacks with hijacked updates

[129] Kim Zetter, "Flame Hijacks Microsoft Update to Spread Malware Disguised As Legit Code," Wired, June 4, 2012, https://www.wired.com/2012/06/flame-microsoft-certificate/; Radware, "Flame," DDoSPedia, https://www.radware.com/security/ddos-knowledge-center/ddospedia/flame/.

[130] David Maynor et al., "The MeDoc Connection," Cisco Talos, July 5, 2017, https://blog.talosintelligence.com/2017/07/the-medoc-connection.html; "Not Petya," MITRE Att&ck, https://attack.mitre.org/software/S0368/.

[131] Tom Warren, "Hackers hid malware in CCleaner software," The Verge, September 18, 2017, https://www.theverge.com/2017/9/18/16325202/ccleaner-hack-malware-security; Edmund Brumaghin et al., "CCleanup: A Vast Number of Machines at Risk," Cisco Talos, September 18, 2017, https://blog.talosintelligence.com/2017/09/avast-distributes-malware.html.

2.5.2.5 Disposal/Retirement

At the end of its lifecycle, a software product is usually retired. This is either done by the software provider directly, who stops offering and supporting the product, or it occurs on the customer end by replacing it with another software product altogether. Retired software and ICT assets can pose security threats. This is most evident on the hardware side of things when for example flash drives, phones, and hard disks are not properly wiped and data recovery by an outside party is unsuccessful. In this case, there is a risk that thrown-away data storages still retain valuable data, including credentials, personally identifiable information, and credit card details. When it comes to retired software, two types of threats fall into the remit of SSCAs. The first type of attack vector includes the exploitation of unsupported or obsolete software, primarily through un-mitigated vulnerabilities. The second type of attack vector leverages obsolete or neglected software components, such as open-source software dependencies and libraries.

The second type is of particular concern, as the bulk of commercial software and applications contain outdated or abandoned open-source components. According to an audit conducted by the software company Synopsys of over 1,500 codebases, 91% of these had components that were either more than four years out of date or that had not seen any development activity within the past two years.[133] Three quarters of the audited codebases also contained known vulnerabilities of which half were of high risk. Theoretically it is also feasible that neglected libraries and open-source software could be taken over by motivated malicious actors if the original author is inactive or deleted their account. For this study, we could not find any examples in the public domain of supply chain attacks that leveraged these kind of threat vectors.

One interesting edge case that might be related to this threat vector is the story of Marak Squires.[134] Marak was the developer of two popular npm packages known as colors.js and faker.js. colors.js was used to change the

[132] Martin Untersinger, "Piratage de CCleaner : la piste de l'espionnage économique ciblé," Le Monde, September 22, 2017, https://www.lemonde.fr/pixels/article/2017/09/22/piratage-de-ccleaner-la-piste-de-l-espionnage-economique-cible_5189878_4408996.html; Avast, "Additional information regarding the recent CCleaner APT security incident," Threat Intelligence Team, September 25, 2017, https://blog.avast.com/additional-information-regarding-the-recent-ccleaner-apt-security-incident.

[133] "2022 Open Source Security and Risk Analysis Report," Synopsis, https://www.synopsys.com/software-integrity/resources/analyst-reports/open-source-security-risk-analysis.html.

[134] Ax Sharma, "Dev corrupts NPM libs 'colors' and 'faker' breaking thousands of apps," BleepingComputer, January 9, 2022, https://www.bleepingcomputer.com/news/security/dev-corrupts-npm-libs-colors-and-faker-breaking-thousands-of-apps/.

colors and style of a text, and faker.js was used to create fake data. Taken together, these two npm packages registered close to 25 million downloads per week. On 9 January 2022, Marak decided to upload a malicious commit that corrupted both color.js and faker.js, triggering an infinite loop that essentially caused a denial of service. Marak's explained his motivation by noting that he was not willing to work for free anymore and pointed to the problem of monetizing open-source work.[135] While this case is an outlier, Marak's conduct does highlight the risks associated with relying on free open-source dependencies and the potential for disruption and exploitation.

2.6 Targeted Assets

As explained in the previous sections, there are a multitude of assets and artifacts that can be targeted and leveraged by malicious actors to conduct SSCAs. Table 4 provides an overview of potentially targetable assets. Delineating these and understanding the role they play, including their potential vulnerabilities, is a critical step in grasping the broad attack surface that allows SSCAs to be so successful.

Targeted assets	Examples
First- and third-party software	Software that is native to targeted system or used by supplier/customer (For instance, web servers, applications, databases, monitoring systems, cloud applications, firmware, apps, etc.)
Software libraries	Third-party libraries, repositories, dependencies, package managers
Code	Source code or code developed by third parties, and code development tools such as SDKs
Configurations	Passwords, API key, firewall rules, URLs
Data and personal data	Information about suppliers, sensor values, certificates, personal data such as payment data, video, documents, emails, sales, financial data, IP, location, credentials, customer data, cryptocurrency wallets, documentation of internal processes and operations
Processes	Update, support, validation processes, code signing, certificate generation, network monitoring, management platform, data cloud transfer
Bandwidth	to conduct DDoS or spam attacks, as a conduit for further infection
Hardware	Chips, USBs, hard drives etc.
Individuals	People with access to data, infrastructure, and info, such as developers

Table 7: Examples of potentially targetable assets during a SSCA.

Geography also plays a role when attackers conduct supply chain attacks. We can discern between two approaches: SSCAs that are global and SSCAs that are regional or sectoral. The StealthyTrident campaign in 2020 and the SignSight campaign in 2021 are both examples of region-/ sector-specific SSCAs.

In the StealthyTrident campaign, a Mongolian software company named Able, which supplies 430 government agencies in Mongolia, was compromised when an attacker gained access to the company's update servers and forced users to download three different trojanized installers via its Able Desktop chat application.[136] The espionage campaign has not been definitively attributed.

Operation SignSight was a supply chain attack against the Vietnamese Government Certification Authority (VGCA). This campaign occurred only a few weeks after the SSCA against Able in Mongolia. In the VGCA case, the attackers were able to modify two of the software installers available on the VGCA website with a backdoor.[137] According to ESET, the two trojanized installers were not properly signed and neither were the clean VGCA installers either. In fact, the official and the trojanized installers used certificates assigned to the same company: Safenet.[138] As ESET notes in its write-up, "[i]n

[135] Marcus Lucero, "The story behind colors.js and faker.js," Revenera, February 9, 2022, https://www.revenera.com/blog/software-composition-analysis/the-story-behind-colors-js-and-faker-js/.

[136] Mathieu Tartare, "Operation StealthyTrident: corporate software under attack," WeLiveSecurity, ESET, December 10, 2020, https://www.welivesecurity.com/2020/12/10/luckymouse-ta428-compromise-able-desktop/; ENISA, "2021 Threat Landscape for Supply chain attacks," July 29, 2021, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport

[137] ENISA, "2021 Threat Landscape for Supply chain attacks," July 29, 2021, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport; Ignacio Sanmillan and Matthieu Faou, "Operation SignSight: Supply-chain attack against a certification authority in Southeast Asia," WeLiveSecurity, ESET, December 17, 2020, https://www.welivesecurity.com/2020/12/17/operation-signsight-supply-chain-attack-southeast-asia/; Hiroki Hada, "Panda's New Arsenal: Part 3 Smanager," NTT Security, December 11, 2020, https://insight-jp.nttsecurity.com/post/102glv5/pandas-new-arsenal-part-3-smanager.

[138] Ignacio Sanmillan & Matthieu Faou, "Operation SignSight: Supply-chain attack against a certification authority in Southeast Asia," WeLiveSecurity,

the case of the attack in Vietnam, we were not able to recover data about post-compromise activity and thus we don't have visibility into the end goal of the attackers."[139]

2.7 Operational Objectives

The socio-technical impacts of a supply chain attack depend on the perpetrator's end goals, targets, and techniques. As illustrated, SSCAs can be quite diverse in nature and impact. In conjunction with table 5, the following paragraphs provide a non-exhaustive list of SSCA operational objectives. More precisely, the potential effects on the targeted system include:[140]

- Violating confidentiality by intercepting data through unauthorized access.
- Reducing integrity by modifying or fabricating information/data/code, which could cause the system to malfunction; end-users to mistrust the information and information system.
- Reducing availability by degrading or interrupting processes that make the system and information or resource unavailable.

An SSCA can remain discreet and undetected for a long period of time, during which attackers are able to extract the data they seek. This can encompass installing backdoors or gaining remote command execution. Alternatively, an attacker might also potentially deploy a crypto miner or adware on a target system for financial gain or divert financial resources and outright steal cryptocurrencies. Other SSCAs are more direct, more overt, and more disruptive. These types of SSCAs can for example lead to widespread disruption, such as in the case of NotPetya, or lead to physical destruction, as was the case with Stuxnet. Moreover, system disruptions can lead to cascading failures and the exposure of systemic vulnerabilities.

Types	Description	Example
Data Extraction	Attackers extract data for economic or strategic espionage, extortion, data harvesting for sale, or to enable other attacks	SolarWinds/ Sunburst, Equifax, Triton
Physical harm	The attack leads to damage to systems (e.g., industrial control) or harms individuals (including loss of life)	Stuxnet, Ukrainian power grid
Backdoor access	Attackers establish an access point for further compromise	ShadowPad, CCleaner
Remote command execution/ download	Attackers conduct remote code execution	GoldenSpy, RubyGems Backdoor
Adware	Attackers install adware for their own profit or to redirect ad traffic	Fake WhatsApp, Simbad, Soraka
Cryptominer	Attackers install a Cryptominer	MSI Font Double Supply Chain Hack
Payment diversion	Attacker divert payments from victims to themselves; steal cryptocurrency wallet	Joker, BankBot, ExpensiveWall
Botnet	Attackers established a botnet targeting or leveraging the target	SimDisk, VestaCP
Data damage	Attackers encrypt, delete, or hide system data	Kaseya, NotPetya

Table 8: Range of technical and operational impacts

ESET, December 17, 2020, https://www.welivesecurity.com/2020/12/17/operation-signsight-supply-chain-attack-southeast-asia/.

[139] Ibid.

[140] William Heinbockel et al., "Supply Chain Attacks and Resiliency Mitigations," Mitre, October 2017, https://www.mitre.org/sites/default/files/publications/pr-18-0854-supply-chain-cyber-resiliency-mitigations.pdf.

3 Assumptions, Enabling Factors, Trends

Today's economic and political environment is characterized by an increasing convergence of the digital, physical, and biological spheres – often referred to as the fourth Industrial Revolution.[141] As a result, any large organization today is seemingly a software driven one. For instance, most production chains are interconnected and include a digital component, from software on industrial control systems, to word processors and office software suits that enable administrative and management tasks. The same is true for physical supply chains which are increasingly digitalized. At the same time, our social and political lives have increased our dependencies on digital tools, software, and platforms. Governments have also become heavily reliant on software products and digital infrastructure to fulfill their functions.

As this hyper-connectivity and digitalization progresses, so do the security risks that come with them. This includes systemic risks linked to cyber events.[142] As demonstrated by the widespread effects of malicious campaigns like NotPetya and Kaseya, SSCAs are used to exploit these systemic risks. While not all supply chain attacks pose the same risks and have a similarly large impact, they generally have the potential to cause considerable damage given the attack vectors they leverage. Section 3.1 describes the operational advantages SSCAs have. Section 3.2 lays out the factors inherent in the current software ecosystem that enable SSCAs. And section 3.3 details the emerging SSCA trends in adversarial behavior.

3.1 Operational Advantages

Conducting SSCAs offers adversaries several operational and tactical advantages. They are often the path of least resistance; they have a widespread reach; they allow for persistence; they can evade common detection techniques; and they can be used for financial gain. Depending on the techniques used, an adversary can gain access to a considerable number of networks and systems. The potential reach – particularly with SSCAs against popular software products – is quite large. The 2018 npm package event-stream incident, which affected 1,600 other packages and was downloaded on average 1.5 million times per week is a good example of these compounding effects.[143]

These advantages are well understood by cybercriminals and intelligence agencies alike as a means of infecting and affecting the broadest possible audience with their malware. The Kaseya and SolarWinds SSCAs illustrate this perfectly. The SSCA against Kaseya impacted over 1,500 firms, while the SSCA against SolarWinds affected 18,000 customers.

Depending on their end goal, SSCAs do allow for the targeting of specific assets. For example, attackers might identify a specific software product in a prospective victim's environment through open-source intelligence research and reconnaissance operations. An attacker might also compromise many assets before refining their targeting to specific networks and companies. The SSCA against SolarWinds is quite illustrative in this context: While it affected 18,000 customers, the threat actor only focused on approximately 100 specific organizations in their post-infection stage.

SSCAs also allow for a certain degree of persistence. Indeed, many SSCAs inject backdoors into specific software products or networks to maintain the initial foothold. The 2015 Juniper breach, explored in more detail in section 5, is a very informative example in this regard.

---

[141] World Economic Forum, "Global Risks Report 2016," https://reports.weforum.org/global-risks-2016/?doing_wp_cron=1645478655.4371941089630126953125;

World Economic Forum, "The Global Risks Report 2022 17th Edition – Insight Report," https://www3.weforum.org/docs/WEF_The_Global_Risks_Report_2022.pdf.

[142] i.e., attacks or other adverse events affecting an individual component of a critical infrastructure ecosystem that will cause significant delay, denial, breakdown, disruption, or loss of services with impact beyond the originating component. Consequences also cascade into (logically and/or geographically) related ecosystem components, resulting in significant adverse effects to public health or safety, economic security, or national security.

[143] NPM, "Details about the event-stream incident," NPM Blog, November 27, 2018, https://blog.npmjs.org/post/180565383195/details-about-the-event-stream-incident.

Finally, SSCAs are quite attractive as they are rather discreet and able to evade many detection techniques used by both human defenders and security tools. Forged certificates, manipulated software installation files, or updates are suitable for subverting widespread detection measures of classical anti-virus programs.[144] Similarly, SSCAs can easily bypass firewalls. In addition, using legitimate administration tools and stolen credentials, a supply chain attack can go undetected for a long time. As illustrated, the SolarWinds/Sunburst attack remained undetected for over nine months. While it is difficult to accurately determine whether non-detection of SSCAs is a systematic problem, other anecdotal evidence tends to support this assumption, particularly when it comes to SSCAs that target open-source components. For example, 20% of the malicious packages examined by Vu et al. persisted in the npm, PyPI, and RubyGems ecosystems for over 400 days.[145] These long durations likely become possible due to the lack of an efficient and fast mechanism to check malicious code injections in packages that are uploaded to package repositories.

Furthermore, even in the event that an attack is known, and the compromise is detected, the defender's response might face substantial difficulties due to the lack of visibility as to what packages and code is used in the majority of software products and applications. For instance, during the global response to the SolarWinds/Sunburst compromise, companies had to check which SolarWinds Orion version they were using and whether they were affected. Meanwhile, many end users were unsure or unaware that the Orion software was part of the product line-up they were using in-house.[146] This complexity is also evident in the Codecov case. Despite the manufacturer publicly disclosing the compromise, the fact that the compromised script was used by thousands of customers and integrated into various programs made it very difficult to identify who was affected.

Detecting an SSCA compromise upstream is also very difficult for software suppliers and customers. Both have very limited visibility and oversight in terms of the development process, origins of code, and the integrity of update mechanisms. The key players who can prevent the introduction of malicious code into software products are the manufacturers themselves. This point is further explored in the next section.

3.2 Software Supply Chain Ecosystem

SSCAs derive their effectiveness from the abuse of the assumed trust between different actors within the software supply chain ecosystem. This includes open-source developers and maintainers, proprietary software developers, software distributors, IT administrators, and end-customers. Especially the latter ones in this list have been conditioned over time to buy, install, and inherently trust software from reputable vendors.

The onslaught of SSCAs has somewhat undermined and eroded this assumed trust. Actors across the board have now started to become wary of software providers on whom they depend for security updates. While some degree of skepticism is certainly positive, widespread dis- and mistrust across the software ecosystem risks hurting the industry at large. Interestingly, the traditional (i.e., pre-Internet) software ecosystem was a safer and more trusted environment. This was mostly due to certain structural characteristics and bureaucratic layers that prevented some SSCA attack vectors. Many of these characteristics and protective measures have evaporated over time. One example is that software was usually developed in-house and within a semi-enclosed environment, rendering physical and electronic access to the code more difficult. Another example is that IT providers used to own a significant portion of the "stack" – meaning the hardware and software components that made up a given IT product. As a result, IT providers like IBM accrued trust because the assumption was that they would be the only accountable party for any issues that arose in their hardware products and software services. IBM would also oversee the maintenance of its products and monitor what code was implemented into its software.[147] This generally provided the manufacturer with a greater overview and control of the entire supply chain.

While the traditional software ecosystem has not completely disappeared – banks for instance build a lot of their software in-house – it has gradually been replaced. As software usage, needs, and demands have become global, an increasingly decentralized ecosystem has emerged in which commercial off-the-shelf solutions –

[144] Bundesamt für Sicherheit in der Informationstechnik, "Die Lage der IT-Sicherheit in Deutschland 2021," bsi.bund.de, https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Lageberichte/Lagebericht2021.pdf?__blob=publicationFile&v=3.

[145] Ruian Duan et al., "Towards Measuring Supply Chain Attacks on Package Managers for Interpreted Languages," Arxiv, December 2, 2020, https://arxiv.org/pdf/2002.01139.pdf.

[146] Swiss Federal Intelligence Service, "Switzerland's Security 2021," https://www.newsd.admin.ch/newsd/message/attachments/67047.pdf.

[147] Tony Scott, "Supply Chain Cybersecurity – A Report on the Current Risks and a Proposal for a Path Forward," https://www-file.huawei.com/-/media/corporate/pdf/trust-center/supply-chain-cybersecurity.pdf?la=en-us.

including both proprietary and open-source software – have become the new normal. Key factors for this transition have been the low-cost of decentralized solutions, interoperability, and innovation pressures resulting from expanded product features and an increasingly diverse choice of suppliers and vendors.[148] Despite these benefits, the decentralized structure has also increased the risk of cyber threats. Of note in this regard is the ICT/OT reliance on a complex, globally distributed, and interconnected supply chain ecosystem that is long-winding, geographically dispersed, and consists of multiple levels of outsourcing.[149]

3.2.1 Globalization and Outsourcing

Consistent with the globalization of markets, ICT and software supply chains have in the past decades become increasingly global in nature. The old IBM/in-house model has slowly given way to a landscape where processes and hardware and software components are globally sourced and developed by a large and geographically dispersed pool of actors. The current ecosystem is composed of a set of internationally diverse public and private sector entities, which includes acquirers, suppliers, developers, system integrators, external providers, and other ICT/OT-related services. This complexity is compounded by differences in legal and policy procedures and practices that guide research and development, design, manufacturing, acquisition, delivery, integration, operation, maintenance, and disposal of ICT/OT products and services.[150]

ICT components are manufactured in numerous different countries. China is the source of most minor ICT components before they are shipped to other locations for assembly, to eventually be dispersed across the world. These products are sold by resellers or integrators who install and operate these products in a variety of organizations.[151] Thus, the end user of a product is often separated by several degrees of magnitude from the initial manufacturer. Coincidentally, at least half of all software development companies outsource their software development.[152] According to Deloitte, 70% of those that did outsource did so for cost reduction reasons.[153] Meanwhile, over 70% of software developers use free software components from diverse sources in their own software products.[154] The sources of these software components are often located abroad; sometimes, their origin is entirely unknown.[155]

Key elements of the current ICT and software supply chain are decentralization and non-tangible data retention and processes, notably through cloud technology, which adds additional layers of complexity to the supply chain environment. As highlighted by New Zealand's National Cyber Security Centre (NCSC), the utilization of cloud systems makes it harder for organizations to attain holistic visibility into the direct and indirect suppliers present in their own supply chains.[156] Cloud systems can also introduce certain supply chain risks, as they tend to require additional applications and familiarity with services to deploy, manage, and secure the cloud environment. Furthermore, the deployment and consumption model of cloud-based applications has also allowed for the rise of shadow IT products and services – meaning projects that are managed outside of, and without the knowledge of, the in-house IT department which can quickly become embedded into an organization's operational process.

Outsourcing and the global nature of ICT supply chains can also lead to geopolitical struggles and imperatives. It is now undeniable that ICT development and security have become key elements in today's great power competition and intelligence contest. The US-China dispute over Huawei's role in the global build-up of 5G infrastructure served as a vivid reminder of the inherent mistrust with respect to potential hardware and software backdoors in products made in environments that could come under adversary influence. Banning the use of Kaspersky products in US public institutions – out of the fear that the firm might be coopted by Russian state actors – is another illustrative example.

[148] NIST, "PRE-DRAFT Call for Comments: Supply Chain Risk Management Practices for Federal Information Systems and Organizations," SP 800-161 REV.1, February 4, 2020, https://csrc.nist.gov/publications/detail/sp/800-161/rev-1/archive/2020-02-04.

[149] Ibid.

[150] Ibid.

[151] Ibid.

[152] CodingSANS, "State of Software Development 2021," https://codingsans.com/state-of-software-development-2021.

[153] Deloitte, "How much disruption? Deloitte Global Outsourcing Survey 2020," p.6, https://www2.deloitte.com/content/dam/Deloitte/se/Documents/technology/gx-2020-global-outsourcing-survey-how-much-disruption.pdf.

[154] Sonatype, "2019 Software Supply Chain Report," https://www.sonatype.com/resources/white-paper-state-of-software-supply-chain-report-2019.

[155] Barabanov et al., "On Systematics of the Information Security of Software Supply Chains," In: Radek Silhavy et al., "Software Engineering Perspectives in Intelligent Systems," Proceedings of the 4th Computational Methods in Systems and Software 2020, Vol. 1, https://link.springer.com/chapter/10.1007/978-3-030-63322-6_9.

[156] New Zealand National Cyber Security Centre, "Supply Chain Cyber Security," Government Communications Security Bureau, https://www.ncsc.govt.nz/assets/NCSC-Documents/NCSC-Supply-Chain-Cyber-Security.pdf.

3.2.2 Privileged Access etc.

A last set of key factors that enable SSCAs is tied to the inherent function, characteristics, and nature of certain software products, which makes them ideal vectors for an attack. Many software products require a degree of privileged access to function efficiently. Default access and sometimes also admin privileges are often given without much reflection and awareness. Once these system privileges are given to certain software products, their access might not be revoked by a service provider or downstream customer once the use of the software has ended, leaving their privileged access active until removed by a system administrator.[157]

Other software products require or demand frequent interaction with a specific service provider across network boundaries. And while it is required and helpful to receive updates and other notifications, the high degree of constant back-and-forth opens the door to potential exploitation. It is also difficult to detect anomalous system behavior when the normal network traffic with external nodes is already high.

Some software products also operate across different platforms within the same entity. This grants them transversal access which makes them attractive targets for attackers seeking to move laterally. This includes common software products such as: asset management software, antivirus, endpoint protection platforms, network monitoring, centralized policy management, and data loss prevention software.[158]

Similarly, software that acts as a centralized management platform to many or all nodes within an enterprise is also an attractive target and a potent attack vector. This is because these central platforms need administrator or root access to allow them to communicate and control every other device on the enterprise network.[159]

3.3 Evolving Adversarial Behavior

Leveraging SSCAs to conduct cyber campaigns is not new. One of its earliest iterations can be traced back as far as 1974, when a US Air Force red team inserted a backdoor in the MIT's Multics operating system, which was eventually incorporated into Honeywell's master copy before being distributed. What is new, however, is the evolution in behavior and TTPs of malicious actors over the past few years. Three trends can be identified:

1. Malicious actors are increasingly using SSCAs.
2. The sophistication of SSCAs has increased.
3. A wider range of actors have started to leverage SSCAs, including cyber criminals.

3.3.1 Frequency

The first trend is that threat intelligence companies, think tanks, and government agencies have identified an increase in the occurrence of SSCAs since at least 2017. Worryingly, this SSCA trend seems to be accelerating with an increasing number of high-profile cases publicly disclosed. In 2017, seven impactful SSCA incidents were reported. By comparison, during the period of 2014-2016 between four to ten SSCA incidents are known.[160] Between 2020 and 2021, ENISA reported 24 SSCA cases.[161] The Atlantic Council's database shows a similar trajectory with 5 publicly reported SSCAs in 2016, 15 in 2017, and 17 in 2018. However, the number for 2019 stands at 16 and the incomplete dataset for the year 2020 shows a mere 8 SSCAs (the report was published in July 2020).[162]

Industry studies and reports on supply chain cyberattacks over the last five years highlight several interesting

[157] Microsoft, "NOBELIUM targeting delegated administrative privileges to facilitate broader attacks," Microsoft Threat Intelligence Center, October 25, 2021, https://www.microsoft.com/security/blog/2021/10/25/nobelium-targeting-delegated-administrative-privileges-to-facilitate-broader-attacks/.

[158] Daniel West, "Software Supply Chain Targeting – Who will the APTs target next?" Obscurity Labs, July 9, 2021, https://obscuritylabs.com/blog/software-supply-chain-targeting-who-will-the-apts-target-next/.

[159] Ibid.

[160] US NIST-CSRC, "Software Supply chain Attacks", 2017, https://csrc.nist.gov/CSRC/media/Projects/Supply-Chain-Risk-Management/documents/ssca/2017-winter/NCSC_Placemat.pdf.

[161] ENISA, "Threat Landscape for Supply chain attacks," July 2021, p. 21, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport.

[162] Trey Herr et al., "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 25, 2020, p. 6, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/. Microsoft, however reports at least 10 for this period (see "Attack inception: Compromised supply chain within a supply chain poses new risks," Microsoft Defender Security Research Team, July 26, 2018, https://www.microsoft.com/en-us/security/blog/2018/07/26/attack-inception-compromised-supply-chain-within-a-supply-chain-poses-new-risks/; Elia Florio, "The Unexpected Attack vector: software updater," RSA Conference, 2018, https://www.youtube.com/watch?v=vcaylXiEwgw.

facts and statistics. Symantec, for instance reported that SSCAs have increased over 78% in 2018 compared to 2017.[163] This trend has only grown during the pandemic and the continuous thrust toward digitalization. Indeed, ENISA estimated in its threat landscape report in July 2021 that the entire period of 2021 would see at least four times more supply chain attacks than the year 2020.[164] According to Mandiant, supply chain compromises were the second most prevalent initial vectors in 2021 while accounting for 17% of intrusions in 2021 (vs. less than 1% in 2020).[165] The trend was noticed by many key stakeholders which tried to raise awareness about SSCAs. In 2018, Microsoft reported in its annual security intelligence report that SSCAs were one of the key threat vectors for industry and other stakeholders.[166] Microsoft notably already foresaw the proliferation of cloud enabled SSCAs.

One year later, US cybersecurity company CrowdStrike commissioned a global survey which identified that 66% of respondents experienced some form of software supply chain attack against their organization, and 33% explained that they experienced an SSCA at least once within the last 12 months.[167] These numbers are echoed in the 2018 Ponemon Institute report on Data Risk in the Third-Party Ecosystem.[168] Ponemon found that 59% of companies surveyed experienced a data breach caused by a third party. Perhaps more worrying is that many of these campaigns appeared to be successful and caused significant damage. 90% of the companies that responded to the 2019 CrowdStrike survey did end up facing financial impacts as a result of the attacks.

US supply chain security company Sonatype reported in its 2021 State of the Supply Chain report that SSCAs aimed at open-source projects rose by 650% during the course of 2021.[169] Sonatype also underlined that the time required for hackers to exploit a newly disclosed open-source vulnerability has shrunk by 93.5% within the last decade. This indicates that malicious actors have significantly evolved by trying out more attack vectors earlier on in the supply chain, including dependency confusion attacks, typosquatting, or code injection.

Several different explanations have been put forward to explain this trend. As highlighted by the United Nations Institute for Disarmament Research (UNIDIR), the US Computer Security Resource Center at NIST, SINGCERT, and the Swiss Federal Intelligence Service, there seem to be three general reasons for the rise of SSCAs. [170]

The first reason is the lack of proper cyber and process protections throughout the software development and distribution phases. The fact that software products are frequently used by multiple operators makes them a rewarding target not only for criminal organizations but also for state-sponsored actors.

The second reason put forward is that organizations' investment in increased cybersecurity awareness has made other, more well-known, or common avenues of attack less likely to be successful. Other types of threats are becoming less effective or more costly for adversaries to launch.

The third reason is that the pressure to digitalize and implement more telework due to the COVID-19 pandemic has led to a widespread growth in the dependencies on and use of third-party software and services.

3.3.2 Sophistication

The general sense within the literature is that largescale SSCAs will remain the domain of sophisticated state actors. SSCAs leveraged by state actors have become quite commonplace and the level of sophisticated campaigns has increased across the board.[171]

[163] Symantec, "Internet Security Threat Report," February 2019, https://docs.broadcom.com/doc/istr-24-2019-en.

[164] ENISA, "Threat Landscape for Supply chain attacks," July 2021, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport.

[165] Jurgen Kutscher, "M-Trends 2022," Mandiant, https://www.mandiant.com/resources/m-trends-2022.

[166] "Attack inception: Compromised supply chain within a supply chain poses new risks," Microsoft Defender Security Research Team, July 26, 2018, https://www.microsoft.com/security/blog/2018/07/26/attack-inception-compromised-supply-chain-within-a-supply-chain-poses-new-risks/.

[167] Dan Larson, "Global Survey Reveals Supply Chain as a Rising and Critical New Threat Vector," Crowdstrike Blog, July 23, 2018, https://www.crowdstrike.com/blog/global-survey-reveals-supply-chain-as-a-rising-and-critical-new-threat-vector/.

[168] "Opus & Ponemon Institute Announce Results of 2018 Third-Party Data Risk Study: 59% of Companies Experienced a Third-Party Data Breach, Yet Only 16% Say They Effectively Mitigate Third-Party Risks," Business Wire, November 15, 2018, https://www.businesswire.com/news/home/20181115005665/en/Opus-Ponemon-Institute-Announce-Results-2018-Third-Party.

[169] "2021 State of the Software Supply Chain," Sonatype, https://www.sonatype.com/hubfs/Q3%202021-State%20of%20the%20Software%20Supply%20Chain-Report/SSSC-Report-2021_0913_PM_2.pdf?hsLang=en-us.

[170] SingCERT, "The Chain Reaction," May 4, 2020, https://www.csa.gov.sg/singcert/Publications/The-Chain-Reaction; NIST, "Glossary: Logic Bomb," https://csrc.nist.gov/glossary/term/logic_bomb; NIST, "Software Supply Chain Attacks," Winter 2017, https://csrc.nist.gov/CSRC/media/Projects/Supply-Chain-Risk-Management/documents/ssca/2017-winter/NCSC_Placemat.pdf; Swiss Federal Intelligence Service, "Switzerland's Security 2021," https://www.newsd.admin.ch/newsd/message/attachments/67047.pdf.

[171] ENISA, "Threat Landscape for Supply chain attacks," July 2021, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport.

SolarWinds/Sunburst is the prototypical example of such a sophisticated SSCA. The state actor demonstrated impressive operational precision, consistency, discretion, and knowledge of the targeted cloud-based environment.[172] Such TTPs also seem to have attracted the attention of other threat actors. Microsoft's Threat Intelligence Center (MSTIC) and Microsoft's Digital Security Unit (DSU) reported in late-2021 that Iranian threat actors are stepping up espionage campaigns against IT services companies to access their customers' networks.[173] North Korean APT Lazarus has seemingly also been following in the same footsteps.[174]

In terms of TTPs, the most likely distribution vector of an SSCA by a state actor is via the hijacking of the software updating process. Others include the undermining of software certificates, open-source compromise, and mobile app store attacks.[175] These latter distribution vectors are indicative of the complexity and the careful planning and execution that is needed for such operations. As highlighted by ENISA in 2021, the "characterisation of being 'advanced' refers to the whole operation and not necessarily merely to the code. In the end, planning, staging, developing and executing two attacks in two organizations is a complex task."[176]

The rising sophistication and frequency of SSCAs by state actors seem to also stem from state actors learning from their own past attempts, as well as from the failures and successes of other state actors. CrowdStrike co-founder Dimitry Alperovitch for example has argued that the Russian foreign intelligence service (SVR) and its affiliated threat actors have learned from their operational mistakes that led to the fast discovery of their campaign against the White House and other US governmental agencies back in 2014-15.[177] As such, they opted to target the SolarWinds software supply chain in order to gain persistent and stealthier access to the same networks. Chinese and Iranian state actors apparently learned similar lessons from the SVR's behavior.[178]

3.3.3 Diversity of Actors: APTs & Criminals

While SSCAs are usually considered the remit of sophisticated state actors, a look at various SSCA datasets reveals that cybercriminals are also increasingly leveraging SSCAs. The Atlantic Council's dataset shows that nearly a quarter of the 100+ SSCAs registered for the period between 2011 and 2021 were conducted by cybercriminal groups.[179] Some of these criminal groups have shown a degree of sophistication like that of certain state actors. REvil's Kaseya ransomware SSCA campaign is a good example of this level of sophistication. Another example is the 2020 SSCA targeting Accellion. The company's file transfer appliance (FTA) was breached by exploiting multiple zero-day vulnerabilities and its customer records were exfiltrated. Mandiant attributed the targeted campaign to a cybercriminal it tracks as UNC2546.[180]

Past successful SSCA campaigns by both state and cybercriminal actors might further incentivize focused attention on supply chain vulnerabilities and related attack vectors. The profitability and impact of SSCAs has not gone unnoticed in the cybercriminal ecosystem. The ransomware attacks against Colonial Pipeline and JBS are relevant examples for these behavioral changes. Despite the recent uptick in sophisticated criminally motivated SSCAs, most SSCAs fall short of this mark. In other words, cybercriminals tend to focus on lower, less headline grabbing SSCAs that are accessible, easier, and faster to exploit. This includes a variety of attempts against open-source libraries, but also SSCAs leveraged

[172] Trey Herr et al. "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/; CERT-EU, "Threat Landscape Report – Volume 1," June 11, 2021, https://media.cert.europa.eu/static/MEMO/2021/TLP-WHITE-CERT-EU-Threat_Landscape_Report-Volume1.pdf.

[173] "Iranian targeting of IT sector on the rise," Microsoft Threat Intelligence Center (MSTIC) and Microsoft Digital Security Unit (DSU), November 18, 2021, https://www.microsoft.com/en-us/security/blog/2021/11/18/iranian-targeting-of-it-sector-on-the-rise/.

[174] Lisa Vaas, "Lazarus Attackers Turn to the IT Supply Chain," Threat Post, October 26, 2021, https://threatpost.com/lazarus-apt-it-supply-chain/175772/.

[175] Trey Herr et al. "Breaking trust: Shades of crisis across an insecure software supply chain," Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/.

[176] ENISA, "Threat Landscape for Supply chain attacks," July 2021, p. 13, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport.

[177] Dimitri Alperovitch, "Homeland Cybersecurity: Assessing Cyber Threats and Building Resilience," Written Testimony before the US House Committee on Homeland Security, February 10, 2021 https://docs.house.gov/meetings/HM/HM00/20210210/111152/HHRG-117-HM00-Wstate-AlperovitchD-20210210.pdf.

[178] VOA, "Sources: Suspected Chinese Hackers Used SolarWinds Bug to Spy on US Payroll Agency," February 2, 2021, https://www.voanews.com/a/usa_sources-suspected-chinese-hackers-used-solarwinds-bug-spy-us-payroll-agency/6201530.html; Eric Rosenbaum, "Iran is 'leapfrogging our defenses' in a cyber war 'my gut is we lose': Hacking expert Kevin Mandia," CNBC, November 18, 2021, https://www.cnbc.com/2021/11/18/iran-leapfrogging-our-defenses-in-cyber-war-hacking-expert-mandia-.html.

[179] To note, however, nearly half of the attacks within this dataset are not attributed.

[180] Andrew Moore et al., "Cyber Criminals Exploit Accellion FTA for Data Theft and Extortion," Mandiant, February 22, 2021, https://www.mandiant.com/resources/accellion-fta-exploited-for-data-theft-and-extortion; ENISA, "Threat Landscape for Supply chain attacks," July 2021, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport.

against mobile environments for the purpose of installing adware and crypto miners.[181]

4 SSCA Edge Cases

To fully understand the broadness of software supply chain attacks it is necessary to take a deeper dive into specific edge cases, or cases that are not commonly identified as SSCAs or use rare and unique attack vectors. Additionally, this section will further examine policy and practical implications of supply chain attacks. On this basis, this report seeks to facilitate the differentiation of SSCAs from other types of cyberattacks to avoid typological conflation. This section focuses on the following edge cases:

1. Insider threats
2. Legitimate third-party access
3. Web-based compromise
4. Unpatched vulnerabilities

4.1 Insider Threats

From a conceptual point of view, it is not hard to imagine that an employee working on an element within the software supply chain could potentially inject malicious code into a code base. This might also be true for employees working at an MSP, CSP, or any other third-party provider with privileged access to a victim's infrastructure. When it comes to the insider threat aspect of an SSCA, several questions need to be answered. What kind of relationship does the victim and the insider need to have for it to qualify as SSCA? Is an intent to inflict harm a prerequisite? And can the introduction of a known vulnerability that leads to an SSCA be considered part of the SSCA?

In 2008, Juniper was financially and politically pressured to introduce an NSA-written encryption algorithm – known as the Dual Elliptic Curve Deterministic Random Bit Generator (Dual EC DRBG) – in its popular NetScreen firewall device.[182] The use of this algorithm essentially functioned as a backdoor for the NSA to eavesdrop on Juniper's customers overseas as it contained an intentional flaw that the NSA was able to exploit. The backdoor was eventually hijacked and modified in 2012 by an alleged Chinese threat actor (APT 5) who was able to decipher the encrypted data flowing through NetScreen's VPNs. In 2014 APT5 created another backdoor granting them access to Screen OS, the operating system for NetScreen products. During the period of 2012-2014, APT 5 was essentially able to spy on multiple US governmental and military agencies, banks, and telecommunication companies until the campaign was discovered and patched in 2015.

The Juniper breach brings to light several elements pertaining to SSCAs, including that of an insider threat. First, Dual EC DRBG is reminiscent of other attempts – notably the Clipper chip in the 90s – by the NSA to undermine encryption algorithms for the purpose of espionage. Second, while the Chinese cyber espionage element is key to highlighting how the intentional weakening of encryption algorithms can backfire, the role of the NSA in enabling a vulnerability and conducting an insider SSCA is the most relevant element here.

Considering the ongoing efforts to address cyber supply chain risks, it is essential to consider the relationships between tech and software suppliers and the intelligence community in their respective countries. These relationships and security tensions are notably growing due to the increase in great power tensions, fragmentation, and geopolitical competition.

4.2 Legitimate Third-party Access

The first SSCA that was widely covered by media outlets was the December 2013 breach of US retail giant Target.

[181] NIST, "Defending Against Software Supply Chain Attacks," Cybersecurity and Infrastructure Security Agency, April 2021, https://www.cisa.gov/sites/default/files/publications/defending_against_software_supply_chain_attacks_508_1.pdf.

[182] "Wyden, Lee, Booker and 13 House Members Question Juniper Networks Over Secret Government Backdoors," wyden.senate.gov, June 10, 2020, https://www.wyden.senate.gov/news/press-releases/wyden-lee-booker-and-13-house-members-question-juniper-networks-over-secret-government-backdoors; Ron Wyden, "Wyden and Booker Question NSA Response Following Supply Chain Hacks of SolarWinds And Juniper Networks," wyden.senate.gov, January 29, 2021, https://www.wyden.senate.gov/news/press-releases/wyden-and-booker-question-nsa-response-following-supply-chain-hacks-of-solarwinds-and-juniper-networks; Joseph Menn, "Congress seeks answers on Juniper Networks breach amid encryption fight," Reuters, June 10, 2020, https://www.reuters.com/article/us-juniper-encryption-congress-idUSKBN23H2C9; Jordan Robertson, "Juniper Breach Mystery Starts to Clear With New Details on Hackers and U.S. Role," Bloomberg, September 2, 2021, https://www.bloomberg.com/news/features/2021-09-02/juniper-mystery-attacks-traced-to-pentagon-role-and-chinese-hackers.

Many researchers consider this breach to fulfill the ABC of supply chain attacks.[183]

The breach started with a malicious actor successfully spear-phishing one of Target's service suppliers covering heating, ventilation, and air conditioning systems called Fazio Mechanical.[184] Fazio was a convenient entry vector as the company had access to Target's internal network due to them being responsible for remotely monitoring and maintaining the temperature in Target stores across the US. Fazio also had access to Target's Ariba external billing system which is part of the business section of Target's vast network. It is believed that Fazio Mechanical was compromised with a Citadel Trojan which stole Target's credentials. [185]

In the second phase of this breach, the adversary used the stolen credentials to enter Target's network and moved laterally. Due to Target's poor network segmentation practices, the attackers were able to gain access to Target's entire system, including parts of the network that contained sensitive data. The attackers also started to test installing malware onto Target's point of sales devices (BlackPOS), which scanned the memory of the PoS for credit card numbers.

In the third phase, the data was encrypted and moved from the point of sales devices to the compromised internal network systems. During peak hours the credit card information was exfiltrated in bulk to a backdoored server which would push the data to drop sites in Miami and Brazil. The breach resulted in the theft of 40 million credit and debit card details a week before Christmas. Most of the credit card details ended up being sold on underground forums. Target was eventually sued by 47 US states in a class action lawsuit. The company spend 202 million USD on its legal defense and ended up settling for a mere 18.5 million USD.[186]

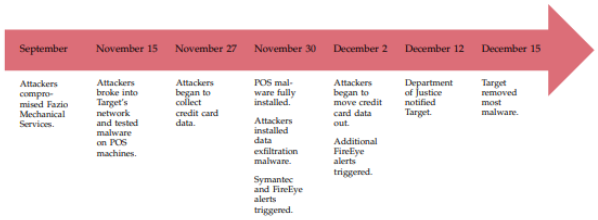

Fig. 1. Timeline of the Target data breach (2013).

Figure 9: Timeline of the Target data breach (Shu, Tian, Ciambrone & Yao, 2017).

The Target breach checks several criteria that makes it a classic SSCA, including two successive breaches, one against the supplier and one against the primary target. Nonetheless, several questions ought to be raised. For instance, is any attack that leverages stolen credentials from a third party to gain access to the end target an SSCA? The likely answer is "it depends." It could be argued that accepting such a broad conceptualization would make SSCAs significantly more common than currently assessed under the existing frameworks. Credential stuffing attacks could also fall into this category. Imagine a scenario in which a hacker buys from access brokers or somehow obtains a database with working credentials – of which some belong to a third parties with legitimate access to other end targets – and the hacker uses those to deliver ransomware payloads, meaning the third-party would not be directly affected by the ransomware campaign. These kinds of attacks are becoming increasingly common and exploit the trusted relationship between the third-party and the end victim.[187] This can, however, be debated, as the said third-party would have to have been the victim of an attack for its credentials to be found in a stolen database: either against them directly or another third-party.

This example raises further questions as to the sequencing of a supply chain attack. Does each phase need to be performed by the same threat actor for the same purpose? And timewise, how long after an attack against a third-party elapse before its access is leveraged against another target and still be considered an SSCA? Does time matter in this context?

[183] "A Brief History of Supply Chain Attacks," Secarma, https://www.secarma.com/a-brief-history-of-supply-chain-attacks/; Kaspersky, "How Target's air conditioning let in a cyberattack," Tomorrow Unlocked, 2022, https://www.tomorrowunlocked.com/supply-chain-attack-target/.

[184] Xiaokui Shu, Ke Tian, Andrew Ciambrone, and Danfeng (Daphne) Yao, "Breaking the Target: An Analysis of

Target Data Breach and Lessons Learned," Arxiv, January 18, 2017, https://arxiv.org/pdf/1701.04940.pdf.

[185] Michael Kassner, "Anatomy of the Target data breach: Missed opportunities and lessons learned," February 2, 2015, ZDNET, https://www.zdnet.com/article/anatomy-of-the-target-data-breach-missed-opportunities-and-lessons-learned/.

[186] Rachel Abrams, "Target to Pay $18.5 Million to 47 States in Security Breach Settlement," The New York Times, May 23, 2017, https://www.nytimes.com/2017/05/23/business/target-security-breach-settlement.html.

[187] Jack Gillum, "Hackers Tried Recycled Passwords on More Than a Million Accounts," Bloomberg, January 5, 2022, https://www.bloomberg.com/news/articles/2022-01-05/hackers-tried-recycled-passwords-on-more-than-a-million-accounts.

4.3 Web-based Compromise

While less discussed in the literature, web based SSCA could be considered edge cases as the level of intermediation between attacker and victim is sometimes very direct (or lacks an intermediary) or doesn't explicitly affect a dedicated phase of the software lifecycle. The UK NCSC, for example, considers traditional watering hole attacks that directly affect an end target via a third-party website a supply chain attack.[188] Three types of such SSCA can be highlighted, illustrating the growing diversity in tactics: traditional watering hole attacks, SSCAs via browser extensions, and SSCAs via website builders.

A watering hole attack is a target approach to compromise a specific organization, group, or individual by infecting websites that those targets frequently visit. Thus, instead of tracking all these targets all the time, an adversary simply infects specific message board or types of websites they know their targets will visit. Watering hole attacks are sometimes also tracked under the attack vector strategic web compromise or are deemed variants of a drive-by-compromise (T1189).[189] These attack vectors can be used for SSCAs. For example, imagine a threat actor who uses a watering hole attack to compromise a website to steal access credentials to breach a software provider, and from there disseminates a malicious update to other end targets.

The UK's NCSC provides the example of the 2012 Voho campaign which coined the term watering hole attack.[190] The Voho campaign was leveraged against government sites, financial services, and websites that promoted democracy in oppressed regions. Other sites connected to political activism, education, and defense in Washington, D.C. and Boston were also targeted. The threat actor compromised these websites and redirected victims to an exploit site that infected them with Gh0st RAT. Gh0st RAT has notably been used by primarily Chinese nation state threat actors. All in all, the Voho campaign was able to redirect 32,000 visitors from 731 unique global organizations and infect almost 4,000 victims with Gh0st RAT (~12% success rate).[191]

Similar in essence are SSCA via browser extensions. An illustrative case is the 2017 Chrome extensions compromise,[192] whereby a malicious actor was able to phish the credentials of developers of several popular chrome extensions (e.g., Web Developer, Infinity New Tab, Web Paint, TouchVPN, and CopyFish) in order to log in, inject a Javascript into the extensions, and push an update. The attack, which affected an approximate 4.8 million users, stole Cloudflare credentials if the victim had a Cloudflare account. It did so by requesting a URL on Cloudflare to get an API key. Once obtained, these were sent to the attacker's website, which then used them to perform several malicious activities, including manipulating internet traffic and serving malicious advertisements that directed victims to affiliate program the attacker profited from.

Browser extensions have remained a popular attack vector despite some efforts by Google and other browsers to ensure their security (e.g., crackdowns, implementing data privacy policy guidelines, and bug bounty programs).[193] In 2019, two researchers[194] found 500 google Chrome extensions that had evaded Google's fraud detection mechanism. Many were copycat plugins of known extensions (e.g., MapsTrek promos), sharing identical functionalities but with a "„.com" added to the name. The extension infected users and exfiltrated private browsing data to conduct malvertising. They would redirect browsers to various domains with advertising streams. While a large portion of these ad streams were actually benign (leading to ads for Macy's, Dell or Best

[188] "Supply chain security guidance," UK National Cyber Security Centre, https://www.ncsc.gov.uk/collection/supply-chain-security/watering-hole-attacks.

[189] "Drive-by Compromise," MITRE Att&ck, https://attack.mitre.org/techniques/T1189/.

[190] "The VOHO campaign: Gh0st RAT spread by water-holing," InfoSecurity, September 26, 2012, https://www.infosecurity-magazine.com/news/the-voho-campaign-gh0st-rat-spread-by-water-holing/; Michael Mimoso, "Large-Scale Water Holing Attack Campaigns Hitting Key Targets," Threat Post, September 25, 2012, https://threatpost.com/large-scale-water-holing-attack-campaigns-hitting-key-targets-092512/77045/; Paul Roberts, "Are Security Firms Ducking Attribution for VOHO? (Rhymes with 'Carolina')," The Security Ledger, October 15, 2012, https://securityledger.com/2012/10/rhymes-with-carolina-are-security-firms-ducking-attribution-for-voho/; Alex Cox et al., "The VOHO Campaign: An In Depth Analysis," RSA FirstWatch Intelligence Report, 2012, https://paper.seebug.org/papers/APT/APT_CyberCriminal_Campagin/2012/VOHO_WP_FINAL_READY-FOR-Publication-09242012_AC.pdf; "The Elderwood Project," Symantec Security Response, September 7, 2012, https://web.archive.org/web/20121011084555/http://www.symantec.com/connect/blogs/elderwood-project.

[191] Michael Mimoso, "Large-Scale Water Holing Attack Campaigns Hitting Key Targets," Threat Post, September 25, 2012, https://threatpost.com/large-scale-water-holing-attack-campaigns-hitting-key-targets-092512/77045/.

[192] Max Maunder, "PSA: 4.8 Million Affected by Chrome Extension Attacks Targeting Site Owners," Wordfence, August 17, 2017, https://www.wordfence.com/blog/2017/08/chrome-browser-extension-attacks/; "Threat actor goes on a Chrome extension hijacking spree," Proofpoint, August 14, 2017, https://www.proofpoint.com/us/threat-insight/post/threat-actor-goes-chrome-extension-hijacking-spree.

[193] Lindsey O'Donnell, "Google Cracks Down on Malicious Chrome Extensions in Major Update," Threat Post, October 2, 2018, https://threatpost.com/google-cracks-down-on-malicious-chrome-extensions-in-major-update/137861/; Chris Brook, "Google Broadens Bounty Program to Include Chrome Extensions," Threat Post, February 5, 2014, https://threatpost.com/google-broadens-bounty-program-to-include-chrome-extensions/104075/.

[194] Jamila Kaya and Jacob Rickerd, "Security Researchers Partner With Chrome To Take Down Browser Extension Fraud Network Affecting Millions of Users," Duo, February 13, 2020, https://duo.com/labs/research/crxcavator-malvertising-2020.

Buy), these legitimate ad streams were coupled with malicious ad streams that redirected users to malware and phishing landing pages. Another more recent case included over 111 malicious Chrome extensions which garnered user downloads and access browser permissions to download malware, keylogging, and general browser surveillance.[195]

A last interesting type of web-browser based SSCA is the man-in-the-browser attack, which include attacks against website builders. This type of attack is somewhat akin to SSCA which compromise SDKs. A key case presented here is the Shylock[196] banking trojan, which up until 2014 targeted e-banking websites in the UK, Italy and the USA.[197] The Shylock attackers were able to compromise legitimate websites through website builders used by creative and digital agencies. They then employed a redirect script, which sent victims to a malicious domain owned by them, which downloaded and installed the trojan onto the systems of those browsing legitimate websites.[198]

4.4 Vulnerabilities

It is important to underline the conceptual difference between vulnerable and malicious (third-party) code. Vulnerable code may include design flaws or errors that are accidentally introduced due to negligence but without any bad intentions. Malicious code is introduced with the intent to cause harm.

The difference comes into play when trying to discern what constitutes a supply chain attack and what does not. Some consider the exploitation of a code vulnerability in a third-party software to be a supply chain attack. The 2017 Equifax hack is often cited as example to underpin this argument.[199] In the Equifax hack the adversary exploited a publicly disclosed but not yet patched vulnerability in the Apache Struts open-source software. Equifax, a US-based credit card reporting agency, had been using Apache Struts as its website framework for systems handling credit disputes. The exploit allowed the adversary to gain access to Equifax's databases exfiltrating a host of PII, including names, birth dates, addresses, driver licenses, and social security numbers of an estimated 143 million Americans.[200] It also compromised the data of up to 44 million British and 8,000 Canadian citizens.[201] The Equifax hack lasted 76 days before the intruder was discovered and the vulnerability fixed.

Despite the Equifax hack encompassing a third-party element in the form of the vulnerable Apache software, it is debatable whether this hack constitutes an SSCA. Its inclusion would expand the SSCA label to a myriad of other incidents including the discovery of the serious Heartbleed vulnerability in OpenSSL back in 2014, and a vulnerability discovered in 2021 in the widely used Log4J library that allowed remote code execution. It would also stretch the SSCA definition to the exploitation of any other known vulnerability if it was leveraged against a downstream target.[202]

ENISA has put forward a similar position by explicitly arguing that "not everything is a supply chain attack."[203] According to ENISA's framework, discovered but non-used software vulnerabilities categorized as non-intentional errors cannot be considered supply chain attacks. Along the same line of reasoning, ENISA argues that libraries and dependencies which are targeted with malicious packages that do not end up being used cannot be regarded as supply chain attacks (but potentially attempts). Thus, they do not consider (at the time of this writing) the previously mentioned use of dependency confusion by Alex Birsan to be a supply chain attack. Nor do they consider the malicious uploads on RubyGems back in December 2020 a supply chain attack.[204]

---

[195] Joseph Menn, "Exclusive: Massive spying on users of Google's Chrome shows new security weakness," Reuters, June 18, 2020, https://www.reuters.com/article/us-alphabet-google-chrome-exclusive/exclusive-massive-spying-on-users-of-googles-chrome-shows-new-security-weakness-idUSKBN23P0JO.

[196] "Shylock," BAE Systems, July 11, 2014, https://www.baesystems.com/en/cybersecurity/feature/shylock.

[197] "'Shylock' malware hit by authorities," BBC, July 10, 2014, https://www.bbc.com/news/technology-28245598.

[198] "Supply chain security guidance," National Cyber Security Centre, https://www.ncsc.gov.uk/collection/supply-chain-security/website-builders.

[199] Matt Howard, "The 'Big Hack' That Actually Happened - Chinese Military Implicated in Equifax Breach," Sonatype, February 11, 2020, https://blog.sonatype.com/chinese-military-implicated-in-equifax-breach.

[200] Todd Haselton, "Credit reporting firm Equifax says data breach could potentially affect 143 million US consumers," CNBC, September 7, 2017, https://www.cnbc.com/2017/09/07/credit-reporting-firm-equifax-says-cybersecurity-incident-could-potentially-affect-143-million-us-consumers.html.

[201] Alex Hern, "Equifax told to inform Britons whether they are at risk after data breach," The Guardian, September 8, 2017, https://www.theguardian.com/technology/2017/sep/08/equifax-told-to-inform-britons-whether-they-are-at-risk-after-data-breach; Vjosa Isai, "Canadians among 143 million people affected in Equifax hack," Toronto Star, September 7, 2017, https://www.thestar.com/business/2017/09/07/equifax-says-data-breach-may-affect-143-million-people-in-us.html.

[202] Risk Based Security, "Is the Kaseya Hack Actually a Supply Chain Attack?" July 14, 2021, https://web.archive.org/web/20220311015932/https://www.riskbasedsecurity.com/2021/07/14/is-the-kaseya-hack-actually-a-supply-chain-attack/.

[203] ENISA, "Threat Landscape for Supply chain attacks," July 2021, p. 26, https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks/@@download/fullReport.

[204] Lawrence Abrams, "Malicious RubyGems packages used in cryptocurrency supply chain attack," BleepingComputer, December 16, 2020,

5 Policy and Practical Implications

Despite numerous articles, marketing pitches, and policy initiatives trying to tackle SSCAs over the years, it took the impact of the SolarWinds/Sunburst campaign to capture the public's imagination as to what SSCAs are and what threat they represent (see figure 10). In recent years, however, the term SSCA has become both overused and is being intentionally conflated with many other types of cyber incidents. In essence, SSCA has become a sort of meta-term, which is being used for a wide range of threat and attack vectors.

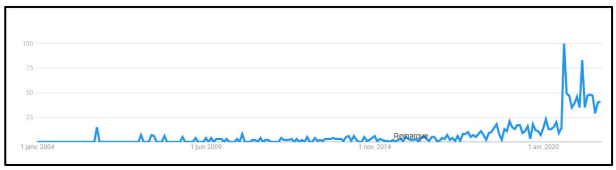

Figure 10: Google Trend search for "supply chain attack" across the world

While the overuse of a particular term usually subsides over time, it can in certain instances become part of a persistent discourse that significantly shapes public perception and epistemological frameworks, thus negatively affecting everything from policy, legislation, and threat models. It is therefore relevant to reflect as to whether (a) the term SSCA is sufficiently specific, and (b) whether the term actually matters.

The answer to the first question is a slippery slope. On the one hand, it can be argued that the term SSCA and its diverse meanings are helpful because they are simple, ambiguous, and vague enough to encapsulate a wide range of disparate issues. Encompassing a rich landscape of risks and vulnerabilities, this allows for a wide range of (policy) actors to address SSCAs and discuss high-profile cases. For instance, in the US, the use of SSCA as a communicative shorthand in the policy discussions emanating from SolarWinds/Sunburst has pushed forward cyber security best practices, standardizations, and the implementation of cyber supply chain risk management solutions across public and private institutions. It has also reinvigorated the debate around a software bill of materials (SBOM) and other legislative instruments that might be helpful in mitigating some of the underlying risks that facilitate SSCAs.

On the other hand, it can be argued that a focus on ambiguity and a wide scope comes at the expense of the technical precision that is required to facilitate better incident response and threat awareness across the first-responder community. Indeed, if everything becomes a supply chain attack, then what exactly is a supply chain attack and how can it be effectively mitigated?

The second question is best addressed by separating it into two parts: Why does naming matter? And does the use of the term *software supply chain attack* beyond the public discourse actually blur or undermine threat mitigation?

Does naming, classification, and terminology matter? Yes, because cyber threat intelligence matters. CTI is usually seen as "evidence-based knowledge, including context, mechanisms, indicators, implications, and actionable advice, about an existing or emerging menace or hazard to assets that can be used to inform decisions regarding the subject's response to that menace or hazard."[205] Accordingly, threat intelligence is key in recognizing, identifying, assessing, and monitoring existing and emerging threat vectors in a more efficient manner. As such it allows users to implement countermeasures and precautionary measures to protect their systems and information. One of the key assumptions underpinning CTI is that it helps maximize the allocation of security investments. Threat intelligence is thus ever more important as cyber threats increase in scope, diversity, complexity, sophistication, and speed, hence making defending more and more challenging for organizations.

To be efficient, CTI relies on threat classification and measurements. These seemingly improve an organization's ability to understand a threat and control it accordingly. Or, as the engineer James Harrington likes to underline, "measurement is the first step that leads to control and eventually improvement. If you can't meas-

https://www.bleepingcomputer.com/news/security/malicious-rubygems-packages-used-in-cryptocurrency-supply-chain-attack/.

[205] "Definition: Threat Intelligence," Gartner Research, May 16, 2013, https://www.gartner.com/en/documents/2487216/definitionthreat-intelligence.

ure something, you can't understand it. If you can't understand it, you can't control it. If you can't control it, you can't improve it."[206]

The process of classification is thus key in organizing, assessing, and evaluating security threats and their impacts, as well as developing strategies to prevent and mitigate them. Classification, however, can be complex and painstaking. Over the past decade, the cyber and information security field has been engaged in countless discussions and initiatives around security incident classification and taxonomies.[207] In the EU, for instance, there are several sometimes overlapping taxonomies, including the Common Taxonomy for Law Enforcement and CSIRTs,[208] the eCSIRT.net taxonomy,[209] the eCSIRT.net mkVI taxonomy, and the Cybersecurity Incident Taxonomy.[210] This is similar, if not worse, in the academic field, where a confusing array of cyber-threat classification systems have been proposed over the past two decades.[211] Some of these are specific frameworks pertaining to SSCAs as shown at the start of this report.

To be relevant and practical, incident taxonomies and frameworks need to be clear. However, the terminology pertaining to SSCAs is, more often than not, lacking nuance and remains ambiguous. As such, it can impede response mechanisms in at least three ways:

1. By blurring, poisoning, and distorting open-source threat intelligence collection.
2. By complexifying threat intelligence analysis and information exchanges.
3. By impeding consensus building around mitigation strategies.

At the practical level, CTI data and insights are usually compiled from a diverse range of sources, such as the National Vulnerability Database, social media platforms, darknet forums, honeypots, and other telemetry.[212] While some of these data collection methods are kept confidential, many CTI firms rely on automated open-source collection methods, for example by scanning and aggregating open-source data by keywords. Accordingly, one could surmise that the unqualified overuse of the term SSCA might potentially distort the perception, metrics, and overall discourse on cyber threats.
Furthermore, international and regional cooperation between public and private CSIRTs and security operation centers (SOCs) have become the norm today, allowing organizations to leverage collective knowledge, experience, and capabilities arising from a community that tries to gain a comprehensive understanding of the threat landscape.

These community exchanges help contribute to early warnings and quick responses as they allow organizations to make threat-informed decisions. However, as the need for information exchange and incident reporting pathways increases, a widely shared, harmonized, and agreed taxonomy is needed.[213] A global harmonization effort that clearly delineates what a software supply chain attack is and what it is not, is a foundational necessity for the expanding threat intelligence ecosystem.

Finally, there is also a policy coordination element to the discussion of getting the definition of SSCAs right. Immature and unaligned classification methods might prevent technical staff, organizational leaders, and policymakers from engaging in a meaningful and nuanced conversations about the threats they face.[214] A loosely defined SSCA terminology and taxonomy might skew the

---

[206] Will Kaydos, *Operational Performance Measurement: Increasing Total Productivity* (Boca Raton, FL: St. Lucie Press, 1999). Quoted in Mark Mateski et al., "Cyber Threat Metrics," Sandia National Laboratories, March 2012, p. 9, https://irp.fas.org/eprint/metrics.pdf.

[207] See: C. Harry and N. Gallagher, "Classifying Cyber Events: A Proposed Taxonomy," Journal of Information Warfare, Vol. 17, No. 3 (Summer 2018), pp. 17-31, https://www.jstor.org/stable/26633163.

[208] Europol, "Common Taxonomy for Law Enforcement and The National Network of CSIRTs," Europol & ENISA, December 2017, https://www.europol.europa.eu/cms/sites/default/files/documents/common_taxonomy_for_law_enforcement_and_csirts_v1.3.pdf.

[209] Don Stikvoort, "Incident Classification / Incident Taxonomy according to eCSIRT.net – adapted," March 2015, https://www.trusted-introducer.org/Incident-Classification-Taxonomy.pdf; ECSIRT, "Appendix C - Incident Classification," http://www.ecsirt.net/cec/service/documents/wp4-clearing-house-policy-v12.html#HEAD6.

[210] EU NIS Cooperation Group, "Cybersecurity Incident Taxonomy," CG Publication 04/2018, July 2018, https://ec.europa.eu/information_society/newsroom/image/document/2018-30/cybersecurity_incident_taxonomy_00CD828C-F851-AFC4-0B1B416696B5F710_53646.pdf.

[211] See: C. Harry and N. Gallagher, "Classifying Cyber Events: A Proposed Taxono-my," Journal of Information Warfare, Vol. 17, No. 3 (Summer 2018), pp. 17-31, https://www.jstor.org/stable/26633163; Mark De Bruijne et al., "Towards a new cyber threat actor topology: A hybrid method for the NCSC cyber security assessment," TU Delft, July 2017, https://repository.wodc.nl/bitstream/handle/20.500.12832/2299/2740_Volledige_Tekst_tcm28-273243.pdf?sequence=1&isAllowed=y; Nils Gruschka & Meiko Jensen "Attack Surfaces: A Taxonomy for Attacks on Cloud Services," 2010 IEEE international conference on Cloud Computing, https://ieeexplore.ieee.org/document/5557984; Jelena Mirkovic & Peter Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," ACM *SIGCOMM Computer Communication Review*, Vol. 34, No. 2, pp. 39-53, https://dl.acm.org/doi/10.1145/997150.997156; Anil Saini et al. "A Taxonomy of Browser Attacks," In: *Handbook of Research on Digital Crime, Cyberspace Security, and Information Assurance*, IGI Global, 2015, pp. 291-313, https://www.igi-global.com/viewtitle.aspx?TitleId=115764&isxn=9781466663244.

[212] Imran Hossen et al., "Generating Cyber Threat Intelligence to Discover Potential Security Threats Using Classification and Topic Modeling," Arxiv, 2021, https://arxiv.org/ftp/arxiv/papers/2108/2108.06862.pdf.

[213] ENISA, "Reference Incident Classification Taxonomy: Task Force Status and Way Forward," January 2018, https://www.enisa.europa.eu/publications/reference-incident-classification-taxonomy/@@download/fullReport.

[214] C. Harry and N. Gallagher, "Classifying Cyber Events: A Proposed Taxonomy," Journal of Information Warfare, Vol. 17, No. 3 (Summer 2018), pp. 17-31, https://www.jstor.org/stable/26633163.

policy solutions put forward. While it is true that there is already a general lack of consensus as to what works best when it comes to cybersecurity, resiliency, and awareness-raising, the lack of clarity and precision in categorizing SSCAs might further contribute to fundamental misunderstandings.

These misunderstandings might in turn lead to the misallocation of scarce resources.[215] Indeed, there are increasingly practical considerations and technical tradeoffs that must be made as to the choice of mitigation measures certain SSCA might require. For instance, patching software vulnerabilities or establishing robust code integrity mechanisms require a very different set of tools and expertise than auditing open-source code and dependencies. While attempting to remedy all these SSCA threats, there is a substantial risk of falling into a so-called "threat intelligence trap," meaning because SSCAs are touching upon so many different elements, a company that tries to guard against everything will fall short on most things.[216] Such risks further underscore the importance for organizations to understand their threat environment. An integrated framework that highlights the SSCA potential of applicable TTPs – rather than compartmentalizing them in an SSCA-specific silo – could overcome the conceptual rigidity of many current taxonomies and inform a more representative mapping of SSCA risks and their relationship with the wider cluster of cyber risks. Data collection at this threshold of SSCA cascades may further sharpen SSCA definitions and the identification of edge cases over time. An integrated understanding of SSCA exposure could thereby guide decisions on security investments and enable the use of defensive resources to greater effect.

[215] Ibid.

[216] Risk Based Security, "Is the Kaseya Hack Actually a Supply Chain Attack?" July 14, 2021, https://web.archive.org/web/20220311015932/https://www.riskbasedsecurity.com/2021/07/14/is-the-kaseya-hack-actually-a-supply-chain-attack/.

6 Annex

Definition	Types/ Author	Source
Supply chain is an ecosystem that includes processes, people, organizations, and distributors dedicated to the development and delivery of a final solution or a product	Academic	Beamon., 1998
The system of organizations, people, activities, information, and resources involved from development to delivery of a product or service from a supplier to a customer. Supply chain 'activities' or 'operations' involve the transformation of raw materials, components, and intellectual property into a product to be delivered to the end customer and necessary coordination and collaboration with suppliers, intermediaries, and third-party service providers.	The MITRE Corporation *(private sector / technology community)*	Deliver Uncompromised – A Strategy for Supply Chain Security and Resilience in Response to the Changing Character of War
Linked set of resources and processes between multiple tiers of developers that begins with the sourcing of products and services and extends through the design, development, manufacturing, processing, handling, and delivery of products and services to the acquirer.	US National Institute of Standards and Technology (NIST) *(government)*	NIST Special Publication 800-53, Rev.5
The network of retailers, distributors, transporters, storage facilities, and suppliers that participate in the sale, delivery, and production of a particular product.	US National Institute of Standards and Technology (NIST) (*government / technology community*)	NIST Special Publication 800-98
Set of organizations with linked set of resources and processes, each of which acts as an acquirer, supplier, or both to form successive supplier relationships established upon placement of a purchase order, agreement, or other formal sourcing agreement. Note 1: A supply chain can include vendors, manufacturing facilities, logistics providers, distribution centres, distributors, wholesalers, and other organizations involved in the manufacturing, processing, design and development, and handling and delivery of the products, or service providers involved in the operation, management, and delivery of the services. Note 2: The supply chain view is relative to the position of the acquirer.	International Organization for Standardization (ISO) / International Electrotechnical Commission (IEC) *(intergovernmental organization)*	ISO/IEC 27036-1:2014

A system of organizations, people, technology, activities, information and resources involved in moving a product or service from supplier (producer) to customer.	European Union Agency for Network and Information Security (ENISA) (*government / technology community*)	Supply Chain Integrity – An Overview of the ICT Supply Chain Risks and Challenges, and Vision for the Way Forward
In general, refers to the whole life of an IT product or service in an organization. It likely includes multiple organizations. Supply chain includes the linked processes of design, manufacture, supply, delivery, support and decommissioning of equipment (hardware and software) or services that are utilized within an organization's cyber ecosystem	Australian Cyber Security Centre (*government / technology community*)	Cyber Supply Chain Risk Management – Practitioners Guide
A set of organizations, people, activities, information, and resources for creating and moving a product or service (including its subelements) from suppliers through to customers.	Open Trusted Technology Forum *(private sector / technology community)*	Open Trusted Technology Provider Standard (O-TTPS)
Linked set of resources and processes between and among multiple levels of enterprises, each of which is an acquirer that begins with the sourcing of products and services and extends through their life cycle.	US National Institute of Standards and Technology (NIST) (*government / technology community*)	NIST Special Publication 800-53, Rev.5; NIST Special Publication 800-161 rev 1.
A supply chain is a set of interconnected processes and resources that starts with the sourcing of raw materials and ends with the delivery of products and services to end users. Supply chains may include producers, suppliers, manufacturers, distributors, wholesalers, vendors, and logistics providers. They include facilities, plants, offices, warehouses, and branches and can be both internal or external to an organization.	International Organization for Standardization *(intergovernmental organization)*	ISO 28000:2007

Annex 1: Supply Chain Definitions (UNIDIR, 2019; complemented by author)

Definition	**Type/ Author**	**Source**
A network of IT infrastructure and technologies that are used to connect, build and share data in virtual networks	Academic	Smith et al., 2007
A cyber supply chain is a complex series of interactions across the lifecycle of all products and services used by an organization.	Australian Cyber Security Centre (*government / technology community*)	Identifying Cyber supply chain Risks
Cyber supply chain is this linked set of resources that can be subject to cyber supply chain risks from suppliers, their supply chains, and their products or services.	US National Institute of Standards and Technology (NIST) (*government / technology community*)	NIST Special Publication 800-161 Rev 1.
E-supply chains involve organizations using online information, to perform, rather than just support, some value-adding activities in the supply chain more efficiently and effectively	Academic	Barlow & Li., 2007

Cyber supply chain is the entire set of key actors and their organizational and process-level interactions that plan, build, manage, maintain, and defend the IT system infrastructure	Academic	Boyson et al., 2009
IT system supply chain is a globally distributed and dynamic collection of people, process, and technology	Academic	Simpson., 2010
A cyber supply chain is a supply chain enhanced by cyber-based technologies to establish an effective value chain	Academic	Kim and Im., 2014
Linked set of resources and processes between acquirers, integrators, and suppliers that begins with the design of ICT products and services and extends through development, sourcing, manufacturing, handling, and delivery of ICT products and services to the acquirer.	US National Institute of Standards and Technology (NIST) *(government)*	NIST Special Publication 800-161
The design, manufacture, delivery, deployment, support and decommissioning of equipment (hardware and software) or services that are utilized within an organization's cyber ecosystem. Supply chain must consider the whole life of an [information technology (IT)] product or service in an organization.	Australian Cyber Security Centre *(government*)	Cyber Supply Chain Risk Management – Practitioners Guide
The manufacturing and/or development process used to produce and deliver hardware or software technology products and their configuration	Open Trusted Technology Forum7 *(private sector / technology community)*	Open Trusted Technology Provider Standard (O-TTPS)
In terms of cybersecurity, supply chain refers to a wide range of software and hardware resources, cloud or local storage, distribution mechanisms such as web applications, and management software. Here are the four key elements of a supply chain: • Supplier: is an entity that supplies a product or service to another entity • Supplier Assets: are valuable elements used by the supplier to produce the product or service • Customer: is the entity that consumes the product or service produced by the supplier. • Customer Assets: are valuable elements owned by the target	European Union Agency for Cybersecurity (ENISA) (*government / technology community*)	Threat Landscape for Supply Chain Attacks
Linked set of resources and processes between acquirers, integrators, and suppliers that begins with the design of ICT products and services and extends through development, sourcing, manufacturing, handling, and delivery of ICT products and services to the acquirer.	US Cybersecurity and Infrastructure Security Agency (CISA) (*government / technology community*)	Defending Against Software Supply Chain Attacks
It captures the network of resources (hardware, software, data, and algorithms) integrated into ICT/OT products and services throughout their entire life cycle. While individuals, organizations, and activities customarily captured in supply chain definitions are excluded here, the inclusion of data and algorithms represents an expansion of traditional supply chain definitions.	Carnegie (t*hink tank*)	ICT supply chain integrity: principles for Governmental and Corporate policies

Annex 2: ICT Supply Chain Definitions (UNIDIR, 2019; and complemented by author)

Definition	Type/ Author	Source
Software supply chain is only one part of a larger, more complex IT solution supply chain.	SafeCode *(private sector / technology community)*	The Software Supply Chain Integrity Framework Defining Risks and Responsibilities for Securing Software in the Global Supply Chain
A system of its participants with an interconnected set of resources and processes involved in the life cycle of software movement from the developer to the end user, namely, design, development, manufacturing, supply, implementation, support of programs and associated services	Academic	Barabanov, Markov, & Tsirlov., 2020

Annex 3: Software Supply Chain Definitions

Definition	Type/ Author	Source
The compromise of a particular asset, e.g., a software provider's infrastructure and commercial software, with the aim to indirectly damage a certain target or targets, e.g., the software provider's clients	European Union Agency for Network and Information Security (ENISA) (*government / technology community*)	Threat Landscape for Supply chain attacks
A supply chain attack is a combination of at least two attacks. The first attack is on a supplier that is then used to attack the target to gain access to its assets. The target can be the final customer or another supplier. Therefore, for an attack to be classified as a supply chain one, both the supplier and the customer have to be targets.	European Union Agency for Network and Information Security (ENISA) (*government / technology community*)	Threat Landscape for Supply chain attacks
A supply chain attack is a cyberattack that aims at damaging an organization by targeting less secure elements in its supply chain.	Cyberpeace (t*hink Tank)*	Playing with Lives: Cyberattacks on Healthcare are Attacks on People
Attacks that allow the adversary to utilize implants or other vulnerabilities inserted prior to installation in order to infiltrate data, or manipulate information technology hardware, software, operating systems, peripherals (information technology products) or services at any point during the life cycle	US Committee on National Security Systems (CNSS) (*government*)	Committee on National Security Systems (CNSS) Glossary, CNSSI 4009-2015
An occurrence within the ICT supply chain whereby an adversary jeopardizes the confidentiality, integrity, or availability of a system or the information the system processes, stores, or transmits. An ICT supply chain compromise can occur anywhere within the system development life cycle of the product or service	US National Institute of Standards and Technology (NIST) (*government / technology community*)	NIST Special Publication 800-161
An intentional malicious action (e.g., insertion, substitution or modification) taken to create and ultimately exploit a vulnerability in Information and Communication Technology (hardware, software, firmware) at any point within the supply chain with the primary goal of disrupting or surveilling a mission using cyber resources.	The MITRE Corporation *(private sector / technology community)*	Supply Chain Attacks and

		Resiliency Mitigations – Guidance for System Security Engineers
An attempt to disrupt the creation of goods by subverting the hardware, software, or configuration of a commercial product, prior to customer delivery (e.g., manufacturing, ordering, or distribution) for the purpose of introducing an exploitable vulnerability.	Open Trusted Technology Forum *(Private sector / technology community)*	Open Trusted Technology Provider Standard (O-TTPS)
A supply chain attack is a type of cyberattack that targets a trusted third-party vendor who offers services or software vital to the supply chain. Software supply chain attacks inject malicious code into an application in order to infect all users of an app.	Crowdstrike (*private sector / technology commnunity)*	What is a Supply Chain Attack
A technique in which an adversary slips malicious code or even a malicious component into a trusted piece of software.	Andy Greenberg (*technology community*)	Hacker Lexicon: What Is a Supply Chain Attack?
A software supply chain attack occurs when an attacker accesses and edits software in the complex software development supply chain to compromise a target farther up on the chain by inserting their own malicious code.	Atlantic Council (t*hink Tank)*	Breaking Trust: Shades of Crisis across an Insecure Software Supply Chain
A software supply-chain attack occurs when attackers target software vendors to compromise legitimate software by introducing malware into their source codes, build processes or update mechanisms.	SingCERT (*government / technology community*)	The Multiplier Effect – Targeting the MSP Supply Chain
A software supply chain attack occurs when a cyber threat actor infiltrates a software vendor's network and employs malicious code to compromise the software before the vendor sends it to their customers. The compromised software then compromises the customer's data or system.	US Cybersecurity andInfrastructure Security Agency (CISA) (*government / technology community*)	Defending Against Software Supply Chain Attacks

Annex 4: Software Supply Chain Attack Definitions (UNIDIR, 2019; and complemented by author)

About the Author

Sean Cordey is a former Researcher in the Cyberdefense Project within the Risk and Resilience Team at the Center for Security Studies (CSS) at ETH Zürich. His research interests include national and European cybersecurity and cyber defense policy, cyber-enabled influence operations, and international cyber law and norms.

CSS
ETH Zürich

The **Center for Security Studies (CSS)** at ETH Zürich is a center of competence for Swiss and international security policy. It offers security policy expertise in research, teaching and consulting. The CSS promotes understanding of security policy challenges as a contribution to a more peaceful world. Its work is independent, practice-relevant, and based on a sound academic footing.