A Software Primer For Attorneys After Cyber Executive Order

By **Alan Charles Raul and Stephen McInerney** (November 22, 2021)

Alan Raul

Stephen McInerney

When President Joe Biden issued his major cybersecurity executive order on May 12, a White House press briefing said the order would invoke

> the power of federal procurement to say, "If you're doing business with us, we need you to practice really good — really good cybersecurity. And, most importantly, we really need you to focus on secure software development."[1]

Over the last six months, Biden's cybersecurity executive order has set in motion a number of prospective changes for the software community that are important for lawyers to understand and monitor.

Although it is not unusual for software vulnerabilities to play a role in cyber incidents, software development has not generally been a significant factor in the legal wrangling over data breaches. There are various reasons for this, including the complexity of analyzing and impossibility of perfecting code, as well as legal protections in software licensing agreements and the usual focus on data owners and service providers.

The inevitability of vulnerabilities and need for ongoing patching and correction have spawned a doctrine known as responsible disclosure. Under this doctrine, for exploitable vulnerabilities that are known only to one or more threat actors — called "zero day" vulnerabilities — software developers and ethical hackers only disclose security-related vulnerabilities or software defects once the problem has been patched or mitigated across all affected releases and versions.[2]

Only then does a software developer publicize the issue and publish the relevant details on common vulnerabilities and exposures and known common weakness enumeration. The reasons for this approach are obvious. Defects are inevitable, defects can lead to vulnerabilities, known vulnerabilities can be exploited, determining whether a vulnerability can be exploited requires developing a proof of concept for a successful attack, and closing exploitable vulnerabilities by fixing defects or other counter measures can take time.

Put another way, software developers do not want to give would-be attackers a treasure map to a new vulnerability until there is a fix. It is therefore not responsible to disclose a vulnerability that is not yet public until it is no longer exploitable by threat actors, as to do otherwise would amplify, rather than minimize, the potential risks from software vulnerabilities.[3]

Software security, which can be contingent on the quality and care reflected in a product's software development life cycle, or SDLC, has recently emerged from relative obscurity. Various high-profile incidents and policy pronouncements have brought greater attention to software security. For example, in the words of the White House, recent supply chain incidents like those at SolarWinds Corp. and Kaseya[4] involved "poor software security."[5]

The National Institute of Standards and Technology also agrees, as stated in a July publication: "Recent incidents have demonstrated the need to better protect the EO — critical software that federal agencies use on-premises, in the cloud, and elsewhere to achieve their missions."[6]

Illustrative of this shift, SolarWinds is currently subject to multiple class actions alleging violations of the federal securities laws and other claims, as well as domestic and foreign investigations and inquiries. In its recent 10-K and 10-Q filings, SolarWinds acknowledged that "it is reasonably possible" it could incur losses associated with the proceedings and investigations.[7]

Moreover, it's also not difficult to imagine a regulator focusing on the reasonableness of a company's software development in determining potential violations or fines after a data security incident, as some regulators, such as the Federal Trade Commission, have already done that in the past.[8]

Also in response, on May 12, Biden issued Executive Order No. 14028, "Improving the Nation's Cybersecurity,"[9] which addresses a variety of issues, such as increased information sharing and the establishment of a Cyber Safety Review Board.

While those other topics are getting much of the attention, Section 4, "Enhancing Software Supply Chain Security," may have the greatest impact on the private sector. The plain language of this section demonstrates its intent to force an improvement in the security of commercial software by leveraging the power of the federal government, particularly its Federal Acquisition Regulation authority, to incentivize more secure software development processes.[10]

Specifically, the order will establish baseline security requirements for critical software, including minimum standards for software verification,[11] require suppliers of software to federal agencies to comply with and attest to compliance with the standards, and catalyze the development of a consumer software labeling program that reflects a baseline level of secure practices.

This shift will certainly require software industry developers to revisit their software development life cycle and existing secure software development processes. Developers will need to assess how the new order will affect requirements and expectations, how the liability landscape may change,[12] and how they can comply with both existing requirements[13] and new initiatives stemming from the order.

This shift will also require lawyers to become better versed in issues related to and diligence regarding software development. Although the order directly applies to government agencies and contractors, one does not need to squint hard to read between the lines and see that standards for all software may be coming soon.[14]

More and more, lawyers will be asked to analyze, on the one hand, software vulnerabilities and development concerns and, on the other hand, reasonable security measures in the wake of a data breach, as defined and conceptualized in various federal and state privacy regimes.[15]

This is, of course, a very broad and far-ranging topic that would require many treatises to fully cover and, in some respects, is just beginning a new chapter in its development — as Biden's order calls for the NIST to issue guidance identifying practices that enhance software supply chain security. The NIST published preliminary guidelines in Appendix F of its publication, "Cybersecurity Supply Chain Risk Management Practices for Systems and Organizations."[16] Final guidelines are due by Feb. 6, 2022.[17]

With that backdrop, the goal of this article is to provide lawyers in particular with an introductory overview of the existing landscape of best practices, standards and guidelines for a secure software development life cycle, or S-SDLC, with an emphasis on software verification; outline Section 4 of "Improving the Nation's Cybersecurity" and its inevitable influence on how corporations approach S-SDLC and software verification; and, lastly, provide a snapshot of recently released guidelines issued by the NIST that recommend 11 minimum practices for software verification.

Current Landscape of Secure Software Development and Software Verification

Secure software development processes date back to the early 2000s, when some software vendors began to integrate specific secure development requirements into their software development processes.[18] Despite this, it is still not common for SDLC models to address software security in detail, and some software development teams may still perceive security measures as an interference.[19]

The executive order and the recent increase in software supply chain attacks, however, have provided an impetus for S-SDLC, which is now poised to become more widely encouraged and adopted. Per the NIST, "SDLC" refers to a "formal or informal methodology for designing, creating, and maintaining software."[20]

More specifically, S-SDLC integrates security measures, including software testing and review, into the development and quality control process. The general goals of security measures in an S-SDLC are to "reduce the number of vulnerabilities in released software, mitigate the potential impact of the exploitation of undetected or unaddressed vulnerabilities, and address the root causes of vulnerabilities to prevent future recurrences."[21]

Although most S-SDLCs share common elements, there is no one-size-fits-all approach. Rather, to be most effective, an S-SDLC must be specific to the software, its environment and the corporation's circumstances. Most SDLC and S-SDLC models and guidelines therefore remain general and flexible enough to apply across development organizations and technologies.

For instance, in connection with its responsibilities under the executive order, the NIST updated its Secure Software Development Framework, or SSDF, which was originally released in April 2020, on Sept. 30, 2021, and discussed the framework during a workshop to receive feedback from thought leaders on Nov. 8. The SSDF recommends a core set of high-level secure software development practices but expressly does not focus "on the tools, techniques, and mechanisms used to" implement the practices.[22]

The SSDF also provides a great reference for established standards, guidance and documents, such as BSA: the Software Alliance's framework for secure software and SAFECode's fundamental practices for secure software development.

In addition to frameworks and best practices, the industry has developed models for assessing and improving an organization's existing SDLC or S-SDLC.

The two most notable are the Open Web Application Security Project's Software Assurance Maturity Model and the Building Security in Maturity Model. Both are technology- and process-agnostic and designed to help organizations develop long-term plans and track progression toward such plans. Thus, players looking to adopt S-SDLC or improve their existing S-SDLC would be well served to look at these models as a good starting point.

Although all elements of a S-SDLC should be evaluated and implemented accordingly, software testing and review[23] may warrant extra scrutiny in light of the order.

Software verification refers to the processes used "to identify vulnerabilities and verify compliance with security requirements"[24] and encompasses many static and active assurance techniques, tools and related processes.[25] Frameworks, guidelines and tools specific to software verification may be more limited than those broadly covering S-SDLC, but they should not be overlooked.

Rather, they should be viewed as complementary and integral to any S-SDLC. In fact, the Open Web Application Security Project's Code Review Guide, which is a comprehensive guide for best practices in secure code review, advises software developers and managers on how secure code review fits within an S-SDLC.[26] And Microsoft Corp.'s Security Development Lifecycle includes three software verification specific practices: static analysis security testing, dynamic analysis security testing and penetration testing.[27]

Effective software verification processes typically include a few common techniques and tools.

First, they analyze the security risks to software by performing threat modeling: "Threat-modeling methods create an abstraction of the system, profiles of potential attackers ... and a catalog of potential threats" that are then used as a basis for testing the software.[28]

Second, they use coding standards, which are collections of coding rules, guidelines and best practices specific to a programming language.

Third, they incorporate scanning and review for known vulnerabilities and software weaknesses. This is generally performed by leveraging the MITRE Corp.'s common vulnerabilities and exposures and common weakness enumeration programs, which are searchable lists of publicly known cybersecurity vulnerabilities and software weakness types.

The scanning and follow-on remediation are often prioritized by using the common vulnerability scoring system and common weakness scoring system programs, which provide a numerical score of the relative severity posed by the vulnerability and weakness, respectively.

Lastly, they incorporate a variety of testing methods, such as static application security testing, dynamic application security testing, interactive application security testing and penetration testing. Although the use of these techniques and tools is currently voluntary, many, if not all, may soon become essentially mandatory because of "Improving the Nation's Cybersecurity."

The Executive Order on Improving the Nation's Cybersecurity

Biden's Executive Order No. 14028, "Improving the Nation's Cybersecurity," followed a series of high-profile cyberattacks, most notably SolarWinds.[29] The incidents highlighted inherent risks related to software security and the flaws in what has been characterized as "the current market development of 'build, sell, and maybe patch later.'"[30] To rectify this state of play, the order, in Section 4, seeks to improve the security of commercial software in three ways.[31]

First, the order requires the establishment of baseline security requirements based on industry best practices. Specifically, Section 4(e) requires the secretary of the U.S. Department of Commerce to issue guidance through the NIST, identifying practices that enhance the security of the software supply chain.[32] The guidance will include standards, procedures, or criteria on ten issues, three of which are directly related to software verification.

Subsection (e)(iii) addresses "automated tools, or comparable processes, to maintain trusted source code supply chains, thereby ensuring the integrity of the code"; subsection (e)(iv) addresses "automated tools, or comparable processes, that check for known and potential vulnerabilities and remediat[ing] them, which shall operate regularly, or at a minimum prior to product, version, or update release"; and subsection (e)(v) requires developers to provide "artifacts of the execution of the tools and processes, described in subsection e(iii) and (iv)" when requested by a purchaser and to make "publicly available summary information on completion of these actions."[33]

This mandate also requires the NIST to "publish guidelines recommending minimum standards for vendors' testing of their software source code."[34] These guidelines were recently published and are discussed in more detail below.

Second, the order uses federal buying power to incentivize the private sector to adopt the guidance by requiring all software purchased by the federal government to meet the standards within nine months. Specifically, subsection 4(k) requires the director of the Office of Management and Budget to "take appropriate steps to require that agencies comply with such guidelines with respect to software procured after the date of this order."[35]

Further, subsections (n) and (o) require recommendations and subsequent amendments, where appropriate and consistent with applicable law, to Federal Acquisition Regulatory Council contract language that will require suppliers of software available for purchase by agencies to comply with, and attest to complying with, any requirements issued pursuant to subsections (g) through (k).[36] And subsection 4(p) requires, as appropriate and consistent with applicable law, the removal of "software products that do not meet the requirements of the amended FAR" from certain government contracts.[37]

Lastly, the order seeks to improve transparency into the software development and verification process. As discussed, subsection 4(e)(v) requires certain disclosures on the software verification tools and processes to purchasers and the public be included in the guidelines on practices to enhance the security of software supply chains. It also requires developers to provide a software bill of materials, which is a list of components in the software, to purchasers.[38]

The order further envisions the development and implementation of a consumer software labeling program that will "reflect a baseline level of secure practices, and if practicable, shall reflect increasingly comprehensive levels of testing and assessment that a product may have undergone."[39] The goal of the labeling program is to "giv[e] the consumer insight while simultaneously rewarding" companies that make devices more secure with recognition in the marketplace.[40]

To that end, on Nov. 1, after receiving input from industry stakeholders, the NIST released draft "Baseline Criteria for Consumer Software Cybersecurity Labeling," which outlines terminology and recommendations for consumer labels — similar to nutrition facts required by the U.S. Food and Drug Administration — for software to more easily describe its security.[41]

For example, information about the software's secure development process, security features of the software, e.g., whether the software is "free from known vulnerabilities" or whether encryption is used), and data stored, processed or transmitted by the software.

The NIST states that the "criteria are intended to aid in the development and voluntary use of labels to indicate that the software incorporates a baseline level of security measures."[42] Comments on the draft criteria are due by Dec. 16, and a final version will be produced by Feb. 6.

Taken together, the requirements in Section 4 of the order signal a potential, significant sea change in the software industry. Corporations and counsel must take account of these requirements; monitor the associated outputs, some of which have already been released; provide input, where appropriate, to influence future outputs like the consumer software labeling program; and implement all applicable guidelines as soon as possible if they envision continuing to supply certain software to the federal government.

NIST's Guidelines on Minimum Standards for Software Verification

Pursuant to the order, the NIST recently released preliminary guidance outlining security measures for critical software[43] and "guidelines recommending minimum standards for vendors' testing of their software source code."[44]

The guidance for security measures for "EO-critical software" is limited to the federal agency use of such software and explicitly does not apply to the "development and acquisition of EO-critical software."[45] It therefore falls outside the scope of this article, but corporations and counsel should consider reviewing it as applicable.

However, the guidelines regarding software verification are directly relevant to S-SDLC and the forthcoming standards that will be required of software developers seeking to sell their products to the federal government.

The "Guidelines on Minimum Standards for Developer Verification of Software" make clear that "[f]requent and thorough verification by developers as early as possible in the [SDLC]" is critical to software security assurance.[46]

The document outlines its scope by clarifying and interpreting terms from the order. Most notably, it expands the definition of "software source code" to include "software in general including binaries, bytecode, and executables, such as libraries and packages"; expands "vendors' testing" to include developers as well; and clarifies its use of "verification" instead of "testing."[47]

It also explains that the document presents minimum standards and should not be used as a guide for the most effective or recommended practices. However, the document does provide supplemental information about verification techniques and references for further information. As required by the order, the NIST recommends implementation of 11 minimum software verification techniques under five general categories.[48]

The five technique classes and eleven recommendations include the following.

Threat Modeling

Do Threat Modeling.

Threat modeling helps identify key or potentially overlooked testing targets, thereby focusing verification on priorities. Threat modeling should be conducted early and done multiple times during development.

Automated Testing

Automate testing as much as possible.

Automated testing can be run often and consistently. It can also check results accurately and minimize the need for human effort and expertise. It can be integrated into an existing workflow or issue tracking system.

Code-Based Static Analysis

Use a code scanner to look for top bugs.

Static application security testing tools can check code for many kinds of vulnerabilities and for compliance with applicable coding standards. For multithreaded or parallel processing software, the NIST recommends using a scanner capable of detecting race conditions.

Review for hard-code secrets.

The NIST recommends using code-based heuristic tools, rather than dynamic testing, to examine the code for hard-coded passwords and private encryption keys.

Dynamic Analysis

Run with built-in checks and protections.

Use built-in checks and protections provided by programming languages during development and in the software shipped.

Create "black box" test cases.

So-called "black box" tests can address functional specifications or requirements, negative tests, denial of service and overload attempts, input boundary analysis and input combinations.

Create code-based structural test cases.

Code-based, or structural, test cases are based on the implementation — the specifics — of the code and may also come from coverage metrics.

Use test cases created to catch previous bugs.

Test cases that have been created to specifically show the presence, or the absence, of a bug can be used to identify issues in the absence of more general first principles' assurance approaches for detecting bugs.

Run a fuzzer.

Automated software testing known as fuzzers can try an immense number of inputs with minimal human supervision. The tools can be programmed with inputs that often reveal bugs.

If the software runs a web service or might be connected to the internet, run a web app scanner.

Use a dynamic analysis security testing or interactive application security testing tool to detect vulnerabilities.

Checking Included Software

Use similar techniques to gain assurance that included libraries, packages, services, etc. are no less secure than your code.

Identify what open source libraries, suites, packages, bundles, kits, etc. the software uses and then use the recommended techniques to ensure the included code is at least as secure as internally developed code.

Conclusion

The nation and software industry appear to be at a crossroads when it comes to cybersecurity and, more specifically, software security. Significant changes within the software industry are coming, and lawyers will play a role at the forefront of these developments. Lawyers must therefore understand existing best practices, standards and guidelines; the requirements imposed on the industry by the order; and how they can ensure and demonstrate compliance with such requirements.

Alan Charles Raul is a partner and Stephen W. McInerney is an associate at Sidley Austin LLP.

The authors thank former Sidley summer associate Philip Robbins and other former summer associates who assisted with the drafting and research for this article.

Disclosure: Sidley Austin and the authors represent numerous software and internet infrastructure and security companies but were not involved in any of the matters discussed in this article.

[1] White House Press Briefing, Background Press Call by Senior Administration Officials on Executive Order Charting a New Course to Improve the Nation's Cybersecurity and Protect Federal Government Networks (May 12, 2021), https://www.whitehouse.gov/briefing-room/press-briefings/2021/05/12/background-press-call-by-senior-administration-officials-on-executive-order-charting-a-new-course-to-improve-the-nations-cybersecurity-and-protect-federal-government-networks/.

[2] Lou Ronnau, Cisco's Process for Fixed Software Release and Vulnerability Disclosure,

Cisco (June 14, 2018), https://blogs.cisco.com/security/ciscos-process-for-fixed-software-release-and-vulnerability-disclosure. In contrast, the full disclosure model advocates for complete vulnerability disclosure immediately. Id.

[3] About CVE, MITRE, https://cve.mitre.org/about/index.html (last updated Mar. 29, 2021); About CWE, MITRE, https://cwe.mitre.org/about/index.html (last updated Mar. 13, 2021).

[4] Isabella Jibilian & Katie Canales, The US is readying sanctions against Russia over the SolarWinds cyber attack. Here's a simple explanation of how the massive hack happened and why it's such a big deal, Business Insider (Apr. 15, 2021, 1:25 PM), https://www.businessinsider.com/solarwinds-hack-explained-government-agencies-cyber-security-2020-12; Charlie Osborne, Kaseya Ransomware Attack: What We Know Now, ZDNet (July 12, 2021, 05:48 AM), https://www.zdnet.com/article/kaseya-ransomware-attack-what-we-know-now/.

[5] White House Press Briefing, Background Press Call by Senior Administration Officials on Executive Order Charting a New Course to Improve the Nation's Cybersecurity and Protect Federal Government Networks (May 12, 2021), https://www.whitehouse.gov/briefing-room/press-briefings/2021/05/12/background-press-call-by-senior-administration-officials-on-executive-order-charting-a-new-course-to-improve-the-nations-cybersecurity-and-protect-federal-government-networks/.

[6] Security Measures for "EO-Critical Software" Use Under Executive Order 14028, NIST (July 9, 2021), https://www.nist.gov/system/files/documents/2021/07/09/Critical%20Software%20Use%20Security%20Measures%20Guidance.pdf.

[7] SolarWinds, Annual Report (Form 10-K) (Mar. 1, 2021); SolarWinds, Quarterly Report (Form 10-Q) (May 10, 2021).

[8] In the Matter of Guidance Software, Inc., FTC Decision and Order, Docket No. C-4187 (2007) (explaining that a company's risk assessment should include a review of its "information systems, including network and software design, information processing, storage, transmission, and disposal; …"); In the Matter of Uber Technologies, Inc., FTC Decision and Order, Docket No. C-4662 (2018) (stating that "the identification of reasonably foreseeable risks" requires a review of a company's "secure software design, development, and testing, including access key and secret key management and secure cloud storage; …"); Federal Trade Commission v. Equifax, Inc., Case No. 19-03297 (N.D. Ga. 2019) (explaining that a company's safeguards for its personal information should include "establishing and enforcing written policies, procedures, guidelines, and standards designed to: (a) Ensure the use of secure development practices for applications developed in-house; …").

[9] Exec. Order No. 14,028, 86 Fed. Reg. 26633, 26633 (May 17, 2021).

[10] See generally id.; Alan Raul et al., Major Executive Order on Cybersecurity Aims to Fortify Defenses and Coordinate U.S. Response to Growing Epidemic of Cyberattacks, Sidley Data Matters (May 14, 2021), https://datamatters.sidley.com/major-executive-order-on-cybersecurity-aims-to-fortify-defenses-and-coordinate-u-s-response-to-growing-epidemic-of-cyberattacks.

[11] Although the EO refers to testing of software source code, NIST uses the term

verification instead. For consistency purposes, this paper will use verification to refer to "testing" within the EO. Paul Black et al., Guidelines on Minimum Standard for Developer Verification Software, NIST (2021), https://www.nist.gov/system/files/documents/2021/07/13/Developer%20Verification%20of%20Software.pdf.

[12] Note that the Federal Trade Commission (FTC) has brought a variety of enforcement actions based on alleged software security issues. These cases resulted in settlements for violations of Section 5 of the FTC Act. For example, in January 2014, TRENDNet reached a settlement with the FTC for marketing statements related to its SecurView Cameras when compared to their faulty software. TRENDNet had allegedly failed to use reasonable security to design and test its software, including a setting for the cameras' password requirement that lead to a compromise and posting of live feeds for nearly 700 cameras. In re TRENDNet, Inc., No. C-4426, 2014 WL 556262 (F.T.C. Jan. 16, 2014). Additionally, in July 2019, D-Link reached a settlement with the FTC related to the company's alleged failure to take steps to address well known and easily preventable security flaws including a software flaw known as a "command injection." FTC Press Release, D-Link Agrees to Make Security Enhancements to Settle FTC Litigation (July 2, 2019), https://www.ftc.gov/news-events/press-releases/2019/07/d-link-agrees-make-security-enhancements-settle-ftc-litigation.

[13] Payment Card Industry – Data Security Standards (PCI-DSS) Requirement 6 currently requires developing and maintaining secure systems and applications. More specifically, PCI-DSS Requirement 6.3.2 requires performing code reviews before applications become active or released to customers. Surkay Baykara, PCI DSS Requirement 6 Explained, PCI DSS Guide (Apr. 7, 2020), https://www.pcidssguide.com/pci-dss-requirement-6/. Additionally, the Federal Financial Institutions Examination Council's (FFIEC) Information Technology Examination Handbook includes a section devoted to "Software Development and Acquisition." FFIEC Information Technology Examination Handbook: Management 31 (FFIEC 2015).

[14] See, e.g., EO 14028 ("[S]hall identify IoT cybersecurity criteria for a consumer labeling program, and shall consider whether such a consumer labeling program may be operated in conjunction with or modeled after any similar existing government programs consistent with applicable law.").

[15] See, e.g., California Consumer Privacy Act, Sec. 1798.100(e) ("(e) A business that collects a consumer's personal information shall implement reasonable security procedures and practices appropriate to the nature of the personal information to protect the personal information from unauthorized or illegal access, destruction, use, modification, or disclosure in accordance with Section 1798.81.5."); Massachusetts Standards for the Protection of Personal Information of Residents of the Commonwealth, 201 CMR 17.03(2)(b) ("Identifying and assessing reasonably foreseeable internal and external risks to the security, confidentiality, and/or integrity of any electronic, paper or other records containing personal information, and evaluating and improving, where necessary, the effectiveness of the current safeguards for limiting such risks . . . .").

[16] See NIST, Cybersecurity Supply Chain Risk Management Practices for Systems and Organizations, Appendix F, available at: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-161r1-draft2.pdf.

[17] See EO 14028, Section 4.

[18] Steven B. Lipner, Standards and Testing for Software Security: Position Paper for the NIST Workshop on Standards and Guidelines to Enhance Software Supply Chain Security 1 (SAFECode 2021).

[19] Donna Dodson et al., Mitigating the Risk of Software Vulnerabilities by Adopting a Secure Software Development Framework (SSDF), NIST 1 (2020), https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04232020.pdf; Synopsys Editorial Team, Secure SDLC 101, Synopsys (July 27, 2020), https://www.synopsys.com/blogs/software-security/secure-sdlc/.

[20] Dodson at 1.

[21] Id. at ii.

[22] See id. at 1.

[23] Black, supra note 11.

[24] Dodson, supra note 19.

[25] Systems and Software Engineering – Software Life Cycle Processes, ISO/IEC/IEEE 12207:2017, https://www.iso.org/obp/ui/#iso:std:iso-iec-ieee:12207:ed-1:v1:en, (last visited July 16, 2021).

[26] See OWASP, Code Review Guide 2.0 (2017).

[27] What are the Microsoft SDL Practices?, Microsoft, https://www.microsoft.com/en-us/securityengineering/sdl/practices (last visited July 26, 2021).

[28] Black, supra note 11 at 5.

[29] Hackers evidently compromised SolarWind's systems and added malicious code into the company's software system "Orion." SolarWinds subsequently sent out software updates to its customers that included the malicious code. Jibilian, supra note 4.

[30] White House Press Briefing, supra note 5.

[31] Id.

[32] Exec. Order No. 14,028, supra note 9, at 26,637-38.

[33] Id. at 26,638.

[34] Id. at 26,640.

[35] Id. at 26,639.

[36] Id. at 26,640-41.

[37] Id. at 26,640. The acquisition actions include "all indefinite delivery indefinite quantity contracts; Federal Supply Schedules; Federal Government-wide Acquisition Contracts; Blanket Purchase Agreements; and Multiple Award Contracts." Id.

[38] Id. at 26,638. On July 12, 2021, NTIA released a document outlining the minimum elements for a Software Bill of Materials. The minimum elements comprise three broad, interrelated areas: data fields, automation support, and practices and processes. Allan Friedman, NTIA Releases Minimum Elements for a Software Bill of Materials, NTIA (July 12, 2021), https://www.ntia.doc.gov/blog/2021/ntia-releases-minimum-elements-software-bill-materials.

[39] Exec. Order No. 14028, supra note 9, at 26,640-41.

[40] White House Press Briefing, supra note 5.

[41] NIST, DRAFT Baseline Criteria for Consumer Software Cybersecurity Labeling (1 November 2021), available at: https://www.nist.gov/system/files/documents/2021/11/01/Draft%20Consumer%20Software%20Labeling.pdf.

[42] NIST, NIST Seeks Public Input on Consumer Software Labeling for Cybersecurity (1 November 2021), available at: https://www.nist.gov/news-events/news/2021/11/nist-seeks-public-input-consumer-software-labeling-cybersecurity.

[43] NIST defines EO-critical software as any software that has, or has direct software dependencies upon, one or more components with at least one of the following attributes: is designed to run with elevated privilege or manage privilege; has direct or privileged access to networking or computing resources; is designed to control access to data or operational technology; performs a function critical to trust; or, operates outside of normal trust boundaries with privileged access. Critical Software – Definition & Explanatory Material, NIST, https://www.nist.gov/itl/executive-order-improving-nations-cybersecurity/critical-software-definition-explanatory (last updated July 9, 2021).

[44] Exec. Order No. 14,028, supra note 9, at 26,640.

[45] Security Measures for "EO-Critical Software" Use Under Executive Order (EO) 14028 – Guidance Purpose and Scope, NIST (last updated July 9, 2021).

[46] Black, supra note 11, at 1.

[47] Id. at 2-3. NIST conceptually defines "verification" as a mental discipline to increase software quality and technically as "techniques, tools, and related processes" used "'to identify vulnerabilities and verify compliance with security requirements." Id. at 1.

[48] NIST also includes one additional task, fixing critical bugs, for completeness purposes. Recommended Minimum Standards for Vendor or Developer Verification (Testing) of Software Under Executive Order (EO) 14028, NIST, https://www.nist.gov/itl/executive-order-improving-nations-cybersecurity/recommended-minimum-standards-vendor-or (last updated July 9, 2021).