Compliance v. Completeness: A Case Study on SBOMs in Consideration of FDA Premarket Cybersecurity Guidance

Logan Kostick
Johns Hopkins University
Baltimore, MD, USA
lkostic1@jh.edu

Michael Rushanan
Harbor Labs
Pikesville, MD, USA
mike@harborlabs.com

Tushar M. Jois
City College of New York
New York, NY, USA
tjois@ccny.cuny.edu

***Abstract*—The U.S. Food and Drug Administration's (FDA) premarket cybersecurity guidance emphasizes the inclusion of a Software Bill of Materials (SBOM) as part of medical device submissions. Although the guidance does not prescribe a specific SBOM standard, it requires manufacturers to provide component data aligned with the National Telecommunications and Information Administration (NTIA) baseline attributes, along with end-of-support dates and support level descriptions. This flexibility is intended to accommodate varying development practices, but can leave critical gaps in risk visibility.**

In this paper, we present a case study of an anonymized medical device SBOM that technically complies with FDA expectations but omits third-party components and dependencies deeply embedded in the software architecture. We also investigate how excluding hardware components exposes devices to overlooked risks. In particular, our analysis shows that expanding an SBOM to include hardware and third-party components increased vulnerability visibility by 18%, finding an additional four critical CVEs in the system. While medical devices may achieve regulatory compliance with an SBOM that includes only the attributes above, such an SBOM does not inherently ensure meaningful cybersecurity risk management.

***Index Terms*—SBOM, computer security, medical devices, hardware, supply chain, regulatory compliance, case study**

1. Introduction

The increasing connectivity of medical devices and their associated software stack has heightened concerns over cybersecurity risks in healthcare environments. The U.S. Food and Drug Administration (FDA) acknowledges this, emphasizing the role of a Software Bill of Materials (SBOM) in medical device cybersecurity since its 2018 draft guidance[1] on premarket submissions [1]. The FDA identified SBOMs as a key tool for managing cybersecurity risks throughout the total product lifecycle, supporting transparency, vulnerability management, and overall device security.

1. The 2018 cybersecurity premarket draft guidance defined the term Cybersecurity Bill of Materials (CBOM), which is analogous to the current use of SBOM.

This position was reinforced through legislation. The 2022 amendments to Section 524B of the Federal Food, Drug, and Cosmetic Act – enacted as part of the Consolidated Appropriations Act, 2023 – formally require that premarket submissions for medical devices with moderate to high patient safety risk include cybersecurity information, such as an SBOM detailing "commercial, open-source, and off-the-shelf software components" [2]. These amendments also strengthen postmarket expectations by requiring "processes and procedures to provide a reasonable assurance that the device and related systems are cybersecure," including during marketing and ongoing maintenance (e.g., software updates and patches).

As a result, medical device manufacturers (MDMs) must create an SBOM for their medical device's software and submit it for regulatory review, as well as provide a plan for monitoring the components therein for vulnerabilities once approved. Failure to do this will result in a blocked premarket submission. More generally, an incomplete SBOM will lead to lower *vulnerability visibility*, i.e., the identification, tracking, and understanding of known security vulnerabilities within software components.

In this paper, we acknowledge that, despite growing regulatory pressure and increased adoption of SBOMs by MDMs, current oversight remains fragmented: characterized by differing interpretations and implementation approaches stemming from the same FDA guidance. As a result, the practice of generating SBOMs often fails to support complete vulnerability visibility and monitoring, due to ambiguities in the process, limitations in tooling, and the inherent complexity of modern software supply chains.

A primary challenge lies in the lack of clarity from existing FDA regulatory guidance regarding what constitutes the *boundaries* of a medical device SBOM. To provide context for this assertion, let us consider a generic medical device product architecture. The architecture may include an embedded system, a mobile device and application, and a backend server that exists within a public server. This product architecture is commonly known as a *system-of-systems*: a product comprised of discrete components that together perform a clinical function, as shown in Fig. 1.

This architecture, although simple on paper, may have a complex hardware and software interactions. The embedded

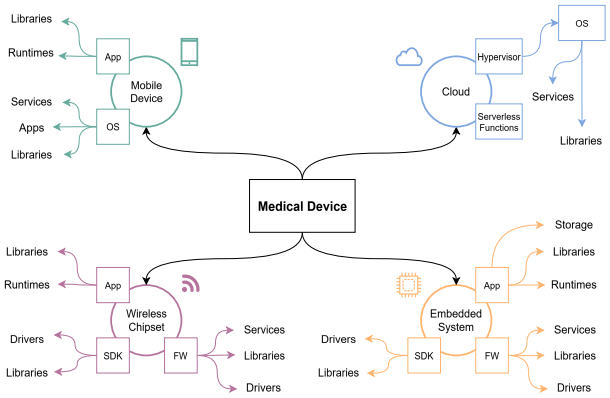

Figure 1: An architecture of a medical device as a system-of-systems of software and hardware components.

system may include various chips to perform analog signal acquisition (e.g., collecting electrocardiogram (ECG) signals), an RTOS to manage processes, ensuring predictable and timely task execution, and a clinical application that implements the UI and clinical functions (e.g., ECG digitization and file I/O). The embedded system may have a separate chip to enable wireless connectivity (e.g., Bluetooth Low Energy) with a mobile device, allowing it to connect to another clinical application. This separate chip may be an off-the-shelf (OTS) component, developed by another vendor and complete with its own software development kit (SDK) and drivers. This SDK may then be used to collect data from the embedded system and then transmit it to a cloud backend over the Internet for aggregation.

An MDM must precisely determine where to draw its boundary for creating its SBOM. Is it the embedded system application and the libraries that comprise it? Is it the OS and the software collectively? Is it the drivers? Is the firmware for the chips? Is it the mobile application and its libraries? Is it the cloud microservices and the underlying software that comprise it (e.g., virtual machine hypervisor and/or serverless interface)? The answer is non-trivial, given the subtle complexity of the architecture.

It remains ambiguous whether SBOMs should be limited to medical application-specific software or whether they must also enumerate operating systems, firmware, drivers, transitive/indirect software dependencies, microservices, and other underlying components. In practice, different interpretations exist.[2] Many MDMs limit their SBOM boundaries to the clinical software they developed and its immediate dependencies. This ambiguity in boundary definition, in turn, significantly impacts vulnerability visibility. Where one draws the boundary can drastically affect what vulnerabilities are discoverable and thus what risks are manageable. In our case study, we demonstrate that expanding the SBOM scope to encompass third-party devices and embedded firmware reveals additional critical vulnerabilities.

2. Anecdotally, we have encountered at least one FDA submission where a reviewer specifically requested a "cloud SBOM" that included the full cloud software stack, further illustrating the ambiguity around scope.

Another challenge stems from practices standard in the open-source software ecosystem. Forking and modifying libraries is a routine process, often resulting in renamed, re-versioned, or repackaged components. These forks may be functionally similar to their upstream counterparts but become unrecognizable to standard vulnerability scanners that rely on name-version string matching. For instance, a fork of OpenSSL renamed as "MedSSL" would not be matched against known CVEs for the parent library, thereby introducing critical blind spots in vulnerability detection.

In the remainder of the paper, we explore the practical implications of SBOM completeness in the context of FDA regulatory expectations, highlighting how current guidance and tooling fall short in supporting adequate vulnerability visibility and risk monitoring. We identify critical gaps that arise from ambiguous software boundaries as well as the challenges of tracking open source and embedded components. Through a case study, we demonstrate how expanding the SBOM scope to include hardware, firmware, and third-party components reveals a considerable increase in observable vulnerabilities. We then provide a discussion of the practical implications revealed through our case study.

2. Background and Related Work

A Bill of Materials (BOM) is a detailed report of the components and processes required for a physical product, gaining popularity in the mid-20th century as manufacturing rapidly evolved [3], [4]. Today, in manufacturing, BOMs are ubiquitous and heavily relied upon in regulated industries, such as medical devices, for certification and component traceability [5]. The concept has been increasingly applied to the software engineering domain, in the form of an SBOM, as an effort to track components of more complex software.

2.1. Software Bill of Materials

In its most recent guidance, Cybersecurity in Medical Devices: Quality System Considerations and Content of Premarket Submissions (June 2025), the FDA outlines specific expectations for SBOMs, or "... a formal, machine-readable inventory of software components and dependencies, information about those components, and their hierarchical relationships" [6]. The agency mandates that component information align with the NTIA's baseline attributes and agency-specific unique fields to provide a comprehensive view of the software supply chain and its potential cybersecurity risks over time [7]. The baseline attributes and definitions are presented in Table 1.

With only descriptive guidance, various SBOM formats have emerged to support these objectives. Three prominent formats include SPDX, developed by the Linux Foundation; SWID tags, standardized by ISO/IEC 19770-2; and CycloneDX, created by OWASP [8]. The divergent nature of the most common formats, combined with ambiguity in the guidance, can lead to confusion about how to properly generate a compliant SBOM.

TABLE 1: SBOM Base Attributes [6], [7].

Attribute	Definition
Author Name	author of the SBOM
Timestamp	time of SBOM creation
Supplier Name	name of the component supplier
Component Name	name of the component
Version String	version of the component
Component Hash†	cryptographic hash of the component
Unique Identifier	unique identification of the component
Relationship	association between SBOM components
Level of Support*	support provided by the original software developer
End-of-Support Date*	expiration of the support provided to this component

† The guidance by the NTIA or the FDA does not specify what the component's hash should consist of, such as the source code or the compiled binary.
* FDA-suggested additional fields.

The literature highlights additional concerns with SBOMs. Pointedly, practitioners worry that the use case for SBOMs will be limited to compliance purposes rather than being a useful tool in understanding the software composition and security posture of a system [8]. Williams et al. note the disparity between SBOM generation and consumption, with an overwhelming focus on the generation of SBOMs [9]. To further complicate matters, a recent large-scale analysis of the accuracy of Java open source SBOMs shows that over 50% of the more than 25,000 analyzed SBOMs contained some sort of inconsistency [10].

2.2. Beyond Clinical Software Vulnerabilities

Medical devices are more than their dedicated medical hardware and clinical software. SBOMs that solely focus on the clinical software while ignoring the computing infrastructure that supports the rest of the system lead to an incomplete analysis of the potentially vulnerable components. While clinical software is a critical component and needs to be secure, auditing the surrounding software and hardware (e.g., operating system, drivers, firmware) also plays an essential part in maintaining medical device security.

A notable example of non-clinical software impacting the security of medical devices is the SweynTooth family of vulnerabilities, which affected Bluetooth Low Energy (BLE) protocol stacks located in vendor-provided driver libraries [11]. SBOMs that solely focused on clinical software, as is the norm, would have missed visibility into this vulnerability class. Perhaps recognizing this lack of visibility, the FDA requested that MDMs perform a risk assessment of their devices using BLE as a precautionary measure [11].

The building blocks of a medical device increase the device's attack surface because any one of these components could lead to partial or complete control of the underlying system. Even a partial compromise of the system can lead to an inability to trust the clinical functionality, potential disclosure of personal health information, loss of availability (e.g., denial of service of the medical device), or, at worst, patient harm. An incomplete, partial SBOM increases the likelihood of missing a vulnerable component.

3. Case Study

To explore the gap between regulatory SBOM compliance and effective cybersecurity risk management, we conducted a case study using the SBOM of a real-world medical device. This device includes built-in sensors to monitor physiological data as well as wireless and local network capabilities that enable communication with desktop applications for remote control and operation.

3.1. Methodology

We now provide the step-by-step methodology used to conduct our case study, presented in the order in which each step was carried out.

SBOM Acquisition. We obtained an SBOM for the medical device from a co-author affiliated with Harbor Labs, a medical device cybersecurity consultancy that assists MDMs in achieving regulatory approval.

The SBOM was formatted according to the OWASP CycloneDX version 1.4 standard and documented not only the manufacturer's software application, operating system, and firmware, but also OTS networking components. While we assert that this comprehensive inclusion is atypical, as it exceeds regulatory guidance, it was achieved through collaboration with the manufacturer to gain a complete understanding of the device's risk beyond the minimum required for approval. It is also beneficial to our case study, as broader inclusion enabled us to adjust the SBOM boundary to encompass varying combinations of software and hardware elements.

SBOM Anonymization. Our case study is based on the SBOM of a real-world medical device. To protect the confidentiality of the MDM and its product, we implemented a Python-based de-identification script that utilizes the CycloneDX Python Library. The script parses arbitrary CycloneDX-formatted SBOMs, in JSON or XML, redacts manufacturer-identifying metadata (e.g., author, name, email, serial number, device type), removes identifying components (e.g., proprietary package names), and updates any component references to avoid dangling pointers (i.e., a reference that points to nothing). It then validates the resultant SBOM against its specification version and generates a new de-identified CycloneDX SBOM file. We created two SBOM variants from this anonymous SBOM:

- Narrow: Included only the software components developed and linked by the manufacturer, including transitive dependencies.
- Extended: Included all elements in the narrow variant, plus embedded firmware, OS, hardware, and OTS devices.[3]

3. The Extended SBOM is the de-identified original—i.e., it includes software and hardware. This approach is atypical and motivates our study.

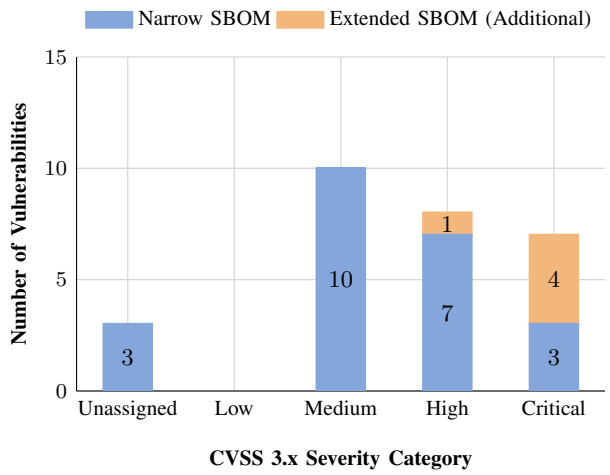

Figure 2: Number of Vulnerabilities by CVSS 3.x Category for the Narrow and Extended SBOMs.

All code and SBOMs from our case study are available as artifacts at https://github.com/spacelab-ccny/sbom-completeness. We also hope to use this repository as a place for the medical device security community to collect and share other anonymized SBOMs.

Vulnerability Analysis. We set up a local Docker container running the open-source OWASP Dependency-Track tool to analyze our Narrow and Extended SBOMs. This tool automatically checks for known vulnerabilities (CVEs) by using Common Platform Enumeration and Package URL values for each SBOM component.

In addition to this automated scan, we manually reviewed hardware and OTS components, since these are often not fully covered by the National Vulnerability Database (NVD). For hardware-related vulnerabilities, such as those in silicon, microcode, drivers, and software tooling, we relied on security bulletins published by manufacturers like ARM and Ubiquiti, which commonly use these bulletins to disclose cybersecurity issues.

3.2. Results

A comparative analysis between the Narrow and Extended SBOMs revealed an 18% change in component coverage and 21% increase in associated vulnerability visibility, as seen in Fig. 2. The Extended SBOM included 10 more components than the Narrow SBOM, an expansion primarily resulting from the addition of embedded firmware, operating systems, hardware devices, and OTS components. However, this increase was not simply "ten new hardware additions." New hardware components often imply new software components. As a concrete example, the inclusion of the Ubiquiti EdgeRouter X brought with it the EdgeOS firmware—demonstrating how expanding the SBOM boundary captures both physical and additional software artifacts.

As expected, the broader scope of the Extended SBOM also revealed more vulnerabilities. Specifically, we identified five additional vulnerabilities compared to the Narrow SBOM. While four of the vulnerabilities were specific to FreeRTOS and automatically determined by Dependency Track and its NVD lookups, one vulnerability was discovered through manual analysis.

The team determined that another OTS component, the Ubiquiti UniFi Access Point, is potentially vulnerable to CVE-2020-11225, a flaw in the Qualcomm WLAN firmware that could allow remote code execution or denial-of-service through malformed wireless frames. They discovered this vulnerability via accessing the device directly and a publicly documented teardown; these teardown images identify the presence of a Qualcomm Atheros QCA9563 Wi-Fi SoC [12], which is known to be vulnerable in this way.[4]

Another insight emerged from the analysis of the Ubiquiti EdgeRouter X running EdgeOS version 2.0.9-hotfix.7. If the device had been running an earlier version, such as 2.0.9-hotfix.5, the number of detected vulnerabilities would have increased by seven [13], representing a 24% increase in vulnerability visibility. This finding highlights the potential impact of not defining a comprehensive SBOM boundary; had they not considered the EdgeRouter within scope for the SBOM, the MDM may not have realized they could be running a highly vulnerable version of EdgeOS.

4. Discussion

Our findings suggest a disconnect between regulatory guidance and device reality. Modern medical devices encompass not only the clinical components of the device but also the entirety of its supporting infrastructure, including both hardware and software. We discuss further below.

OTS Components. The OTS system found in our study, the Ubiquiti Access Point, is not part of the clinical software; yet, it is a critical component for the device's communication scheme. Because regulatory guidance does not consider these non-clinical components to be included in the SBOM, the FDA and the MDM lose one of the important aspects of this exercise: vulnerability management for a marketed device. Not only was vulnerability visibility lost in this instance, but because the FDA guidance does not require all OTS components to be included in the SBOM, the OTS ecosystem is left untracked.

This is especially a problem *postmarket*, i.e., after the device is approved and available. A vulnerability in an OTS component could lead to a vulnerability in the core medical device – without any visibility, as the postmarket surveillance tools are monitoring an incomplete SBOM. Thus, although preparing an SBOM is a premarket activity, a more complete one would lead to more effective postmarket surveillance.

Approaching the generation of an SBOM as an *SBOM-of-SBOMs*, that is, each component provides an SBOM that

4. We communicated with Ubiquiti to confirm the vulnerability. Ubiquiti informed us that the device is not vulnerable to the CVE and does not invoke the vulnerable functionality described in CVE-2020-11225.

is included in the system's SBOM, would significantly enhance usability. In our study, the team had to manually identify the chipset and firmware of the access point. However, if an SBOM was provided by the access point manufacturer, the team could have simply added the component's SBOM to the system. Fortunately, CycloneDX supports the concept of "Component Assemblies." Furthermore, the CycloneDX format provides extensibility to be used for a variety of purposes beyond SBOMs. CycloneDX refers to these as *x*BOM and has capabilities to support physical inventorying for a Hardware Bill of Materials or Manufacturing Bill of Materials. The format also supports specialized objects, such as the generation of a Cryptography Bill of Materials or a Machine Learning Bill of Materials [14]. Thus, one could imagine tooling that supports the automatic collection and inclusion of SBOMs for each OTS component when the system SBOM is generated.

Software Derivatives. As our study shows, maintaining and documenting each component in the system as part of the SBOM is key to vulnerability management; however, in the open source development community, many libraries or programs are forked and renamed. This renaming of software removes the ability for tools such as Dependency Track to determine if a component is vulnerable. To check if a component has a vulnerability, Dependency Track queries the vulnerability database for an entry with both the software name and version for CVEs. By changing one or both parts of the query, the lookup fails. The SBOM used in the study included FreeRTOS as one of its components; consider, however, a scenario where the SBOM listed a FreeRTOS derivative such as OPENRTOS instead [15]. In this case, the vulnerability would be missing; indeed, a brief search of the NVD as of this writing shows *zero results* for OPENRTOS vulnerabilities [16], while there are 24 results for FreeRTOS [17], despite OPENRTOS essentially being a re-licensed FreeRTOS [15]. Thus, under current tooling, MDMs *must* use the "correct" name for a dependency, lest they lose all visibility into it. Indeed, this highlights a broader problem: the tools that we use to ensure the security of safety-critical medical devices are exceedingly brittle.

Vulnerability, Exploitability, and Visibility. Our case study highlights how a more complete SBOM provides for a more comprehensive understanding of a medical device's vulnerabilities. However, the presence of a vulnerability in software does not automatically mean it is exploitable in a given environment. A vulnerability is a potential weakness, but whether it poses actual risk depends on several contextual factors, e.g., intended use, operating environment, configuration, user behavior, and compensating controls. But *visibility* into vulnerabilities enables an MDM to determine whether their systems are susceptible to an attack and how to mitigate the associated risk. *Not* having this visibility means that an actual exploit may catch an MDM by surprise.

5. Conclusions and Future Directions

In this paper, we present a case study that highlights the importance of extending SBOMs of medical devices beyond the core clinical software included on-device. From this case study, we draw the following lessons for the different stakeholders in medical device security:

Industry. The biggest takeaway for industry participants such as MDMs is that compliance with FDA cybersecurity guidance does not indicate the completeness of the product's cybersecurity mitigations. The guidance should be taken as a baseline; MDMs should strive to go beyond it to achieve a more comprehensive view of their product SBOMs, allowing in turn for better vulnerability visibility.

A secondary takeaway concerns how OTS components should be integrated with medical devices. As we show in our case study, adding the OTS router to the medical device SBOM improved vulnerability visibility. But, adding the router was a *manual* process which required analysis of both its software and hardware. Thus, we recommend that OTS vendors create SBOMs for their products, and MDMs request them when choosing an OTS component. Market factors may encourage OTS vendors to do so: SBOMs help MDMs with vulnerability visibility, helping mitigate security risks faster – which, in turn, is good for business. MDMs may prefer OTS vendors that offer SBOMs with their components, as these vendor-provided SBOMs reduce the cybersecurity analysis burden of the MDM.

Regulators. Current guidance from regulators, such as the FDA, does not address the inclusion of OTS hardware components that include software in SBOMs. Given the results of our case study, we recommend that regulators update their guidance to recommend including OTS components in SBOMs. There may be pushback from MDMs, as an increase in regulatory requirements will necessarily lead to more resources being invested in compliance, resulting in a slower time-to-market. Regulators also have to contend with larger SBOMs, which means lengthier review times for premarket submissions. With the complexity and interdependencies of modern medical devices, however, we assert that such a step is necessary for better cybersecurity risk management and patient outcomes.

We believe that while our case study is instructive, it is not a systematic view of the SBOM landscape for medical devices. An assessment that expands the scope of this one case to many others would be helpful in drawing better conclusions on where the boundary of what should and should not be included in a medical device SBOM. Given regulators' view into *all* medical device SBOMs, we believe they are the right stakeholders to perform this assessment. A potential goal for a broader assessment would be to identify standard components that are more likely to be vulnerable and exploitable. Since there is a tension between SBOM usability and completeness, such an activity would help optimize the review efforts of regulators as well as the preparation efforts of MDMs.

Academics. The fragmented SBOM space contains several research questions for academics to pursue. In addition to the steps discussed above to further assessment, we suggest further exploration of the tooling used to generate SBOMs. One concrete direction is to develop automated methods

to integrate SBOMs into the medical device development lifecycle. With our case study in mind, such automated methods should be able to operate not just on software and package managers (e.g., as identified by Stalnaker et al. [8]), but also on lower-level components such as firmware and drivers. Another direction could be to analyze the landscape of forks of software, and how SBOMs can adapt to handle forked software in sensitive components (e.g., by integrating fork integrity analysis [18]).

Healthcare Providers. In this paper, we focus on the technical aspects of medical devices, but we argue that there are takeaways for the "end users" of medical devices as well. Healthcare providers act as the gatekeepers for medical devices for their patients, and insecure devices can represent a risk to their patients, potentially equivalent to that of an incorrect dosage [19]. Thus, when deciding on a medical device-based therapy, we recommend that healthcare providers ask the MDM for information about the device's cybersecurity. While we do not expect clinicians to ask for full SBOMs or cybersecurity disclosures, our case study highlights some basic questions they can ask an MDM before choosing to adopt their product. What outside components or infrastructure does the device require? How does the MDM manage the cybersecurity of the device along with its outside components? The more healthcare providers ask these sorts of questions, the more MDMs are willing to invest in better security and transparency.

Patients. We argue that patients deserve the same transparency as healthcare providers so they can make an informed decision as to choice of medical device and MDM. But more generally, at the end of the day, cybersecurity controls for medical devices impact real people. So, achieving completeness, rather than just compliance, must be the goal.

Acknowledgments

The authors would like to thank the anonymous peer reviewers of HealthSec '25 for their helpful comments. The authors would also like to thank Nick Yuran from Harbor Labs for his support and feedback, and Luis Vargas and Devang Jain from Harbor Labs for their help with analyzing the original SBOM in this case study.

This work is supported by National Science Foundation (NSF) awards 1955172 and 2451597, a Sui Foundation Academic Research Award, PSC-CUNY awards, and awards from Google as a part of the Cyber NYC program.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors. Any mention of specific companies or products does not imply any endorsement by the authors, by their employers, or by the sponsors.

References

[1] U.S. Food and Drug Administration, "Content of Premarket Submissions for Management of Cybersecurity in Medical Devices," Oct. 2018. [Online]. Available: https://web.archive.org/web/20181103014648/https://www.fda.gov/downloads/MedicalDevices/DeviceRegulationandGuidance/GuidanceDocuments/UCM623529.pdf

[2] U.S. Congress, "Consolidated Appropriations Act, 2023," Dec. 2022.

[3] P. S. Rusk, "The role of the bill of material in manufacturing systems," *Engineering Costs and Production Economics*, vol. 19, no. 1, pp. 205–211, May 1990.

[4] J. G. Miller and L. G. Sprague, "Behind the Growth in Materials Requirements Planning," *Harvard Business Review*, Sep. 1975.

[5] Cybersecurity and Infrastructure Security Agency, "A Hardware Bill of Materials (HBOM) Framework for Supply Chain Risk Management," Sep. 2023.

[6] National Telecommunications and Information Administration, "Framing Software Component Transparency: Establishing a Common Software Bill of Materials (SBOM)," Oct. 2021. [Online]. Available: https://www.ntia.gov/sites/default/files/publications/ntia_sbom_framing_2nd_edition_20211021_0.pdf

[7] U.S. Food and Drug Administration, "Cybersecurity in Medical Devices: Quality System Considerations and Content of Premarket Submissions," Jun. 2025. [Online]. Available: https://www.fda.gov/media/119933/download

[8] T. Stalnaker, N. Wintersgill, O. Chaparro, M. Di Penta, D. M. German, and D. Poshyvanyk, "BOMs Away! Inside the Minds of Stakeholders: A Comprehensive Study of Bills of Materials for Software Systems," in *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. Lisbon Portugal: ACM, Feb. 2024, pp. 1–13.

[9] L. Williams, G. Benedetti, S. Hamer, R. Paramitha, I. Rahman, M. Tamanna, G. Tystahl, N. Zahan, P. Morrison, Y. Acar, M. Cukier, C. Kästner, A. Kapravelos, D. Wermke, and W. Enck, "Research Directions in Software Supply Chain Security," *ACM Trans. Softw. Eng. Methodol.*, vol. 34, no. 5, pp. 146:1–146:38, May 2025.

[10] Y. Xiao, D. Kirat, D. L. Schales, J. Jang, L. Xing, and X. Liao, "JBomAudit: Assessing the Landscape, Compliance, and Security Implications of Java SBOMs," in *Proceedings 2025 Network and Distributed System Security Symposium*. San Diego, CA, USA: Internet Society, 2025.

[11] U.S. Food and Drug Administration, "FDA Informs Patients, Providers and Manufacturers About Potential Cybersecurity Vulnerabilities in Certain Medical Devices with Bluetooth Low Energy." [Online]. Available: https://www.fda.gov/news-events/press-announcements/fda-informs-patients-providers-and-manufacturers-about-potential-cybersecurity-vulnerabilities-0

[12] OpenWrt Project, "Ubiquiti unifi ac [openwrt wiki]," 2025, accessed: 2025-07-23. [Online]. Available: https://openwrt.org/toh/ubiquiti/unifiac

[13] NIST, "NVD - search results for cpe:2.3:o:ui:er-x_firmware:2.0.9," 2025, accessed: 2025-07-23. [Online]. Available: https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&isCpeNameSearch=true&seach_type=all&query=cpe:2.3:o:ui:er-x_firmware:2.0.9:-:::::::

[14] The OWASP Foundation, "Authoritative Guide to SBOM," Apr. 2024. [Online]. Available: https://cyclonedx.org/guides/OWASP_CycloneDX-Authoritative-Guide-to-SBOM-en.pdf

[15] "OPENRTOS®, part of embedded FreeRTOS – OpenRTOS® – SafeRTOS family." [Online]. Available: https://www.highintegritysystems.com/openrtos/

[16] NIST, "NVD vulnerability search: OPENRTOS," accessed: 2025-07-26. [Online]. Available: https://nvd.nist.gov/vuln/search#/nvd/home?keyword=OPENRTOS&resultType=records

[17] NIST, "NVD vulnerability search: FreeRTOS," accessed: 2025-07-26. [Online]. Available: https://nvd.nist.gov/vuln/search#/nvd/home?keyword=FreeRTOS&resultType=records

[18] A. Cao and B. Dolan-Gavitt, "What the Fork? Finding and Analyzing Malware in Github Forks," in *Proc. of NDSS*, vol. 22, 2022.

[19] D. Klonoff and J. Han, "The first recall of a diabetes device because of cybersecurity risks," pp. 817–820, 2019.