SBoM Vulnerability Assessment & Corresponding Requirements

L Jean Camp
Vafa Andalibi
ljean@ljean.com
vafandal@iu.edu
Luddy School of Informatics, Computing, & Engineering
Bloomington, IN, USA

In this comment we provide a brief overview of the vulnerabilities present in the Software Bill of Materials proposal at is currently stands. We address the standard as they are publicly documented in the summer of 2021. We focus on systematic classes of vulnerabilities building on the minimal Software Bill of Materials (SBoM) standards [7].

The attacks which leveraged the Solarwinds software updating practices dramatically illuminated the previously apparent need for organizations to easily track their exposure and dependencies [2]. The Software Bill of Materials (SBoM) proposal is an immensely important step forward in software supply chains and has the potential to fundamentally change the calculus of risk in software purchasing and operation .

The security of BoM itself is not central to its adoption.

In this work we highlight categories of potential vulnerabilities with the assumption that as soon as SBoMs are in use they will be under attack. We then close with recommendations both in process and contents of the standards.

1 THE MINIMAL SBOM

In this section we identify the components in the minimal SBoM ecosystem. We make explicit trust assumptions that are currently implicit. We focus on the file containing the essential information, often referred to as the BoM.

For each SBoM there is an initial manufacturer of software who is associated with the SBoM file, the BoM. There is a source of the file, which is associated with the manufacturer. That file lists the packages or smaller components in the software, how these are integrated, and either additional information such as licenses or constraints on use. Figure 1 shows a high level illustration of SPDX and CycloneDX, two options for data-files for SBoMs. Following the CycloneDX pattern, we will refer to the datafiles as *BoMs*.

Both CycloneDX and SPDX data structures begin with the creator of the content, the date of content, and of course the identification of the standard and version for use. Each of these is presented as unverified attestations with assumptions embedded about Uniform Resource Identifier (URI). A commonly used resource identifier URI is a domain name, and these are presumably included as identifiers either as contact information or as an identifier of the manufacturer. Therefore the presumption is that the attestation of the manufacturer is in the form of TLS/SSL, with the associated public key certificate.

Information about when and by whom the files are reviewed are contained as annotations in the SPDX and as information about changes or commits in CycloneDX. This structure provides the ability of organizations to make their own attestations presumably as signed linked documents. An example of possible use is an industry-specific Information Sharing and Analysis Center (ISAC) or a set of trusted experts who might determine that the particular SBoM is correct and correctly authenticated. Consider that the Department of Energy (DoE) might provide such attestations for use in DoE facilities.

2 SBOM RISK ASSESSMENT

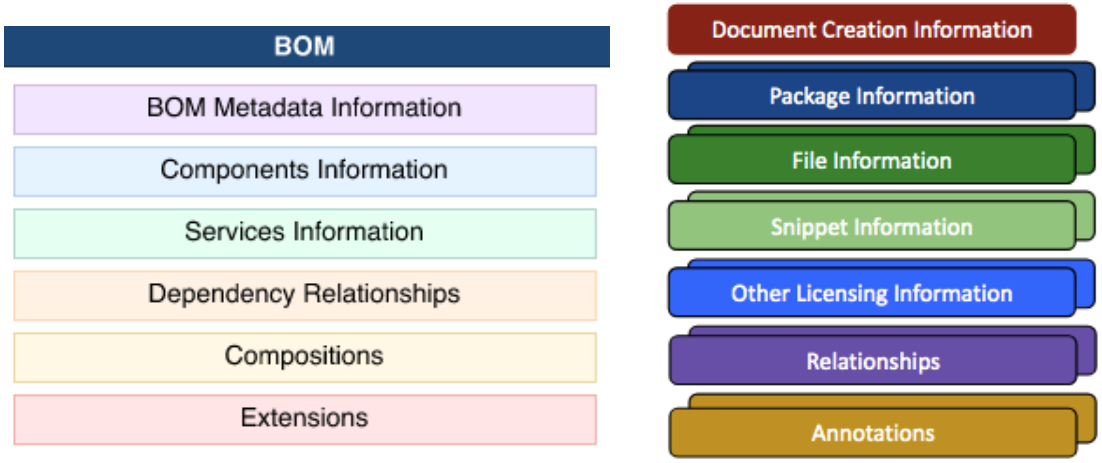
In the previous section we note that there is a need for SBoM availability. For software which can be entirely downloaded at time of purchase, the BoMs would presumably be a component of the downloaded package. For software that is embedded in a cyber-physical device, the integration of the Manufacturer's Usage Description and the BoMs is a natural pairing, and one that is being put forward both from the SBoM Working Group and the IETF. Figure 2 illustrates the process by which a MUD-File, and thus any corresponding BoM, is obtained and installed as part of a secure on-boarding process.

Neither of these include standards for SBoM availability, which is not a solved problem either in practical nor theoretical terms. The ability to target only those entities which are known to be vulnerable with a zero-day on a specific package is an issue that had been identified as a risk in SBoM. The state of the art scanning for services and vulnerabilities means that it is a reasonable to assert that the availability of a SBoM creates little advantage for the attacker. In contrast, it creates significant advantages for attackers by making visible the vulnerabilities and dependencies, as well as the need to patch.

Another reason for availability (and arguably web-based distribution) is for well-managed organizations to obtain BoMs when purchasing or taking over operations from companies that were not as well managed. Limited access with emails that may no longer be maintained or viewed makes archiving much more difficult.

It is also necessary for SBoM to be accessible to every potential customer in order for the value of transparency to be realized in the marketplace. It must be accessible so that organizations seeking to make informed business decisions can do so. For this reason, the choice to allow secretive and limited distributions ((e.g., email as an option for communication of SBoMs) from manufacturers) removes one of the core advantages of SBoMs: making security in software less of a lemons market [2]. Such a policy can result in changes and updates that are difficult to track, enable licensing agreements that prevent researchers and analysts from verifying the correctness of the SBoM with standard reverse engineering practices.

Under the current standard, as proposed by the IETF and as implemented in example projects, both the source of the BoM (i.e., the



(a) CycloneDX Data Structure[1]

(b) SPDX Data Structure [2]

Figure 1: The structures of the two dominant alternatives for SBoM files show the core components addressed in this threat model

MUD-File) not the BoMmay be authenticated, weakly authenticated, or not authenticated simply named. The core assumption appears to be that TLS/SSL will provide adequate source authentication and protect the integrity of the URL and then the content in transit.

The limits of TLS/SSL are not reasonable for all contexts in which BoMs will need to be obtained, used, and trusted. Reliance on TLS/SSL does not provide adequate risk mitigation. It is not radical to assert current PKI governance, by an entity literally named the "CA/Browser Forum" and designed for commercial transactions on the web, may not be optimal for setting standards for attestation about the integrity and source of code in safety-critical and cyber-physical systems. In fact, research shows that current CAs have chronic failures in terms of authenticating providers (CAs) [5], beyond the widespread use of https by phishing and other malicious sites [3]. The current Root Program operators (i.e., Google, Apple, Mozilla, Microsoft) have issued not statements seeking to manage the roots of trust for the SBoM infrastructure.

The other role of TLS/SSL is in authenticating the source of code. This is particularly problematic given that much of the BoMs consists of package names and domain names, and these can change hands. Consider the notorious 11 lines of code which, when deleted, echoed throughout the supply chain [9]. After the deletion, the recovery effort required that package names were then re-appropriated by and to other developers. Of course, domain names can be hijacked as well as package names and repositories, thus reifying the need for stronger authentication.

The inclusion of message digest in the retrieval is not adequate to protect integrity. By definition, MUD and SBoM retrieval would be less common than all other commerce combined, and be directed to dedicated resources. As message integrity for for MUD files and BoMs require collision resistant hash functions (and as noted in the previous paragraph, that are signed independently of the message authentication obtained in TLS).

The current standards for the cryptography for the digests does not address the need that the digests actually verify the contents, as these are not currently collision-free. There is also no post-quantum option in either data standard. Currently, the checksum default is SPDX is SHA-1. Algorithms that can be used include SHA224, SHA256, SHA384, SHA12, as well as, even more unimaginably, MD2, MD3, MD5, and MD6. (Please see section 3.10 in the SPDX standard for the definition of Package Checksum.)

The current standards for digest are adequate for detection of random failures during transmission. However, what is needed is secure SBoM as attackers can either confuse victims into remaining vulnerable or substitute their own packages. After substituting a malicious package, insertion of an update or patch will be greatly simplified. Such attacks have gone from the breakthrough cryptography used in Flame to Solarwinds, and with a flawed weak SBoM standard these could be integrated into future ransomware-as-a-service products (given the protection provided by MD2 for example).

At the most simple level, the cryptographic standards supported should be have strong pre-image resistance and collision resistance. That is for any given SBoM file, it should only be the case that the hash of that file is equal to the hash of another file only if those files are identical. Thus the inclusion of SHA-1 as an acceptable algorithm in CycloneDX and SPDX is difficult to understand, as NIST deprecated the use of SHA-1 in 2011 and it was specifically deprecated in 2013 for digital signatures because of lack of collision resistance [6, 8].

It is clearly not reasonable to expect a universal upgrade of TLS/SSL yet it is equally important and reasonable to have the integrity attestation travel with any BoMs (or MUDFile). Both CycloneDX and SPDX have extensible data structures that would enable these attestations.

The famous 11 lines of code, where a simple utility widely embedded in the infrastructure was deleted illustrates the destruction of

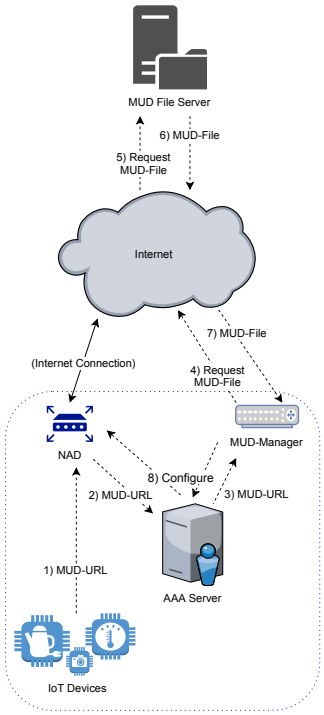


Figure 2: Transferring a MUD file containing either an associated BoMs or the directions on how to obtain it

packages via aggressive legal enforcement [9]. The use of the American legal system as a tool for disruption and information exposure by foreign adversaries has been previously documents [1]. This indicates a need for further research into potential legal threats against SBoM and the MUD infrastructures. Packages and services which embed trademarks should not be subject to removal or conflicts without a significant barrier after the name has been embedded in an SBoM. Addition of a package name should provide specific limited protection for only and exactly use of that name for a package. Trademark excessive enforcement has already cause significant harm to the structure of the Internet, with enforcement against a pre-existing package name. By focusing narrowly on package names, damage from aggressive enforcement of trademark can be mitigated.

SPDX further contains the requirement there be human-readable legal interpretations of any compliance or other legal actions that might need to be taken, and also requires that the file be subject to testing for syntactic correctness. The second - deterministic parsing - can be used to evaluate correctness of SBoM internally, before they are distributed, as well as enable integration of multiple SBoMs for a comprehensive view of an organizations exposure for risk for a specific vulnerability. This requirement should also apply to all BoM data formats. The first requirement - deterministic parsing of legal assertions - is infeasible. Contexts vary according to contracts and context of use (e.g., safety-critical). More broadly terms of use are regularly updated and jurisdictions differ now, with no expectation of universal coordination in judicial decisions.

Finally, there are lessons learned that are valuable for the diffusion and sustainability of SBoM. The SBoM Working Groups are worthy of compliment, and can be strengthened. The process model here could be Mutally Assureed Norms of Routing (MANRS) or the London Accords. This could be concurrent with identification of specific individuals in each industry-specific ISAC. This increase in formalization could also simplify expansion to include additional sector-specific stakeholders, to which we return below.

3 IMMEDIATE REQUIREMENTS

Minimal Standards of Availability

Until issues of conditional availability (presumably using escrow and verification) have been resolved, BoMs should be highly available. The use of email for contacting a distributor of SBoMs is addressed below should be reserved for that future when escrow has been reliably established.

Correct the Inadequate Message Digests

The cryptographic standards used in SBoM should comply with current NIST recommendations, as well as including post-quantum alternatives in order to simply migration to these. A clear source of acceptable algorithms should be identified and when external resources do not meet these standards additional hosting with effective attestations to integrity and source are needed. SHA-1, and even more so explicit message digest standards (MD2, MD5) are dangerously inadequate for critical security information.

Set Cryptographic Requirements for Integrity and Source

Beyond the requirements for the integrity of the core BoM data, there is a need to ensure that contributors remain the same entities adequate to ensure that a takeover or hijack of a package or domain name will not result in the ability to disrupt code updates (or worse to update). As the requirement for accepting the expired Windows 7 and Vista certificates to enable patching illustrates, using cryptography without the ability to chain updates is now widely recognized as a failure to follow best practices. Such a failure in SBoM may create systematic weaknesses in the infrastructure in the future.

Requirements for Use of Email for Distribution

The use of email for contacting a distributor of SBoMs is included in the current MUD/SBOM standard, creating immediate concerns [4]. This implies two requirements: file escrow and email authentication. If an organization chooses to distribute SBoMs via email then there must be a mechanism to obtain escrowed SBoMs as devices and code may long outlive their manufacturers. In the near term, use of secure email must be mandated. Both standards enable access to specific locations in version control systems (e.g., git or svn). However, neither offer the option of including authenticating information in the standard itself; for example, limited use tokens or public key attestations that allow manufacturers to limit access to SBoMs. This is no way implies that passwords or usernames should ever be included; however, methods for secure access to SBoM should be

[1]https://cyclonedx.org/specification/overview/ [Accessed on May 2021]
[2]https://spdx.github.io/spdx-spec/1-rationale/ [Accessed on May 2021]

adopted in place of an option for a specific insecure email address. If this is not adopted in the standard itself, then associated best practices should preclude the practice.

4 FUTURE REQUIREMENTS

Address Risks Created by Law

The need to codify both narrow limits to trademark law to address existing packages and limiting terms of service to ensure third parties can evaluate and verify BoMs is already clear. We need not look for some alternative future where there are legal threats to security and transparency, as these have already occurred. These sources of potential harm should be addressed as soon as feasible.

Inclusion of all Stakeholders

As SBoM matures and pilots are rolled out the scope of stakeholders expands.

Given that availability of SBoMs is a critical issue, funding for the inclusion of Internet Archive or other digital archivists to enable long-term access would enable clear, scientifically grounded best practices for archiving and long-term availability from even suppliers that close or are transient.

In addition, each domain has unique end users. Advocates may be needed for specific groups of end users; for example, patient advocates in medical domains or those engaged in right to repair in transportation.

Specifications of Lifecycles and Revocation Conditions

When ever there is attestation there is a need for revocation. Further work is needed to align the lifecycle and revocation modes with the risk of failures in attestations. Lifecycles of distinct domains may differ significantly in: (a) the role of users in selecting the source of trust, (b) the ability to update a particular attestation or associated software, (c) the ability to remove or update the source of an attestation (e.g., a certificate authority), (d) the expected lifetime of an attestation and (e) the implications of code far outliving its expected lifetime, all of which will be informed by the potential harm from a failure.

Tools for Creation and Visualization of Combinations of BoMs

Adoption of SBoM by developers is an essential part of an effective standard. Simplifying the creation of SBoMs from source code must be made straight-forward. The suggestions above, such as augmenting weak hash functions that do not have strong collision resistance, must be automated for ease of use to be acceptable. On-line tools, training, materials, and potentially the creation of a certification illustrating the mastery of these could be coordinated with industry partners including Stack Overflow, Git, or Google as these are currently the sources of (good and bad) code. Preventing code and guidance that creates vulnerabilities may provide easier that muting and mitigating it after it has been popularized.

Structures for Third Party Verification

Both data structures include in their format potential expansions that could be used for attestations from third parties (e.g., about correctness, reliability, and changes in the status of BoMs). Formalizing structures for these uses can encourage third parties to invest in verification, and for users to be able to share their own verification with other customers across or within industrial sectors.

5 CONCLUSION

The Software Bill of Materials is a promising solution to previously intractable problems of not only securing the supply chain, but even being able to identify its components. The identification of risks in SBoM is not in any way an objection to its adoption or a dismissal of its potential. This comment only notes that as currently written, SBoM risks creating an insecure network management protocols (e.g. Telnet, SNMPv2, FTP) where adoption of integrity attestation and transport security lagged. The potential for SBoM to transform the visibility of risks argues that appropriate attestations should be integrated into the design, and before widespread adaptation. Specific trust assumptions about the names of sources of code should be articulated, and some of these will be more limited than others; for example, use in safety critical systems or military suppliers will meet higher standards than products used for entertainment. For without wide-spread adoption of strong protocol security, SBoM risks being unable to fulfill its promise.

ACKNOWLEDGMENTS

REFERENCES

[1] Alex Finley. 2019. What Durham Is Investigating and Why It Poses a Danger to US Intelligence Analysis. *Just Security* (25 11 2019). https://www.justsecurity.org/author/finleyalex/

[2] Vaibhav Garg. 2021. A Lemon by Any Other Label. In *ICISSP*. 558–565.

[3] Doowon Kim, Haehyun Cho, Yonghwi Kwon, Adam Doupé, Sooel Son, Gail-Joon Ahn, and Tudor Dumitras. 2021. Security Analysis on Practices of Certificate Authorities in the HTTPS Phishing Ecosystem. In *Asia Conference on Computer and Communications Security*. ACM, 407–420.

[4] Eliot Lear. 2020. SBOM Extension for MUD. [Online]. Available on: https://tools.ietf.org/html/draft-lear-opsawg-mud-sbom-00.

[5] Nicolas Serrano, Hilda Hadan, and L Jean Camp. 2019. A complete study of PKI (PKI's Known Incidents), In Telecommunications Policy Research Conference (Washington, DC). *Available at SSRN 3425554*. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3425554

[6] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, and Yarik Markov. 2017. The first collision for full SHA-1. In *Annual International Cryptology Conference*. Springer, 570–596.

[7] National Telecommunications and Information Administration. 2020. Software Bill of Materials . [Online, Accessed on May 24th 2021]. Available on: https://www.ntia.gov/sbom.

[8] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. 2005. Finding collisions in the full SHA-1. In *Annual international cryptology conference*. Springer, 17–36.

[9] Chris Williams. 2016. How one developer just broke Node, Babel and thousands of projects in 11 lines of JavaScript. *The Register* (23 3 2016). https://www.theregister.com/2016/03/23/npm_left_pad_chaos/