Auto-ISAC Software Bill of Materials (SBOM) Informational Report TLP:CLEAR

Auto-ISAC

January 17, 2025

v3.0

Abstract

Findings of the Auto-ISAC SBOM Workgroup for background and effective practices of SBOM usage from the working group proceedings in 2023 and 2024

Contents

List of Figures vi

List of Tables vii

1 Introduction 2
- 1.1 Industry Background . . . 3
- 1.2 SBOM General Background . . . 3
- 1.3 Auto-ISAC SBOM Working Group . . . 3
- 1.4 Automotive SBOM Use . . . 4
- 1.5 Report Objectives and Scope . . . 5
- 1.6 Report Structure and Content . . . 5

2 SBOM Projects in Government, Industry, and Standards 8
- 2.1 Overview . . . 9
- 2.2 United States Government . . . 9
- 2.3 Governments - European Union . . . 12
- 2.4 Governments - Japan . . . 13
- 2.5 Industry . . . 13
- 2.6 Academia, Standards, and Open Source . . . 14

3 Ways the Auto Industry is Unique 18
- 3.1 Introduction . . . 19
- 3.2 Complex Supply Chain . . . 19
- 3.3 Cybersecurity and Safety . . . 20
- 3.4 Updates and Complexity . . . 20
- 3.5 Risk Assessment and Vulnerability Disclosure . . . 20
- 3.6 Cost and Resources Limitation . . . 20
- 3.7 Data Privacy Concerns . . . 21
- 3.8 Autonomous and Semi-Autonomous Technologies . . . 21
- 3.9 Regulatory Compliance . . . 21
- 3.10 Physical Security Threats . . . 21
- 3.11 Interconnected Infrastructure . . . 22
- 3.12 Secure-by-Design Principles . . . 22
- 3.13 Conclusion . . . 22

4 Pre-sale Considerations for SBOMs 24
- 4.1 Introduction . . . 25
- 4.2 Request for Proposal and Request for Quotation . . . 25

4.3 Non-Disclosure Agreement . . . 25
4.4 Cybersecurity Interface Agreement and Development Interface Agreement . . . 26
4.5 Identity and Access Management . . . 27
4.6 Know Your Suppliers . . . 27
4.7 Technology Infrastructure and the Mechanics of Integration . . . 28
4.8 Cryptography Algorithms and Providers . . . 28
4.9 Confidentiality and Intellectual Property . . . 28
4.10 Incident Management and Vulnerability Disclosure . . . 28
4.11 Service Level Agreements . . . 29
4.12 Licensing . . . 29
4.13 SBOM Formats . . . 31
4.14 Vulnerability Exchange and Disclosure Formats . . . 31

5 SBOMs in Development **32**
5.1 Introduction - Software Development Methodologies and SBOM . . . 33
5.2 Actionable versus Complete SBOMs . . . 33
5.3 Automotive Industry Considerations . . . 34
5.4 Software Provenance . . . 34

6 Exchanging and Collaborating with SBOMs **38**
6.1 Introduction . . . 39
6.2 Motivation . . . 39
6.3 Risks . . . 40
6.4 Level of Detail . . . 40
6.5 Stakeholders . . . 40
6.6 Transmission Methods . . . 41
6.7 Integrity Checks . . . 41
6.8 Sharing SBOMs During Product Development . . . 41

7 Vulnerability Operations Using SBOMs **46**
7.1 Introduction . . . 47
7.2 Supply Chain Vulnerability Management . . . 47
7.3 Vulnerability Management - Basic Steps for Using SBOMs . . . 47
7.4 SBOMs and ISO/SAE 21434 Section 8 - Continual Security Activities . . . 49
7.5 Incident Response - SBOM Use . . . 50
7.6 Cautions for SBOM-based Vulnerability Management . . . 51
7.7 Conclusion . . . 52

8 Review of SBOM Workshop Exercises and Findings **54**
8.1 Introduction . . . 55
8.2 Key Insights . . . 55
8.3 SBOM Workshop - Round 1 . . . 55
8.4 SBOM Workshop - Round 2 . . . 57

9 Tools **58**
9.1 Introduction . . . 59
9.2 Automotive Product Development Process - A Prescriptive Method . . . 59
9.3 The ISO 26262 V-Model . . . 59
9.4 Effective Practices . . . 59
9.5 Identity and Access Management . . . 59
9.6 Binary Repository . . . 61
9.7 Source Repository . . . 61

9.8 Messaging / Event-driven / Asynchronous Architecture . . . 62
9.9 Continuous Integration and Continuous Delivery . . . 62
9.10 Vulnerability Scanner . . . 62
9.11 Cryptography Software . . . 62
9.12 SBOM Generation, Validation, Conversion, Reporting and Inventorying . . . 63
9.13 Testing Software . . . 64
9.14 Fuzz Testing (Fuzzing) . . . 65

10 Appendices **66**
A SBOM Examples . . . 67
B Vendors and Tools . . . 77
C Languages and Ecosystems . . . 84
D References . . . 85
E Acronyms . . . 86
F Glossary . . . 90

Acknowledgements

The Auto-ISAC[1] would like to thank the many members of the team that wrote and published this Informational Report ("Report"). Here is a partial list of contributors:

Auto-ISAC Staff

Josh Poster, Director of Intelligence & Analysis
Cassia Green, Support Associate, Coordination and Administration (after October 2023)
Alison Hwang, Coordination and Administration (through October 2023)

Auto-ISAC SBOM Working Group

Chair: Charles Hart, Hitachi America, Ltd.
Vice Chair: Dr. Oliver Creighton, BMW of North America, LLC

Report — Authors

Charles Hart, Hitachi America, Ltd.
Dr. Oliver Creighton, BMW of North America, LLC
Adam Just, Toyota Motor North America (through April 2024) and West Mountain LLC (present)
Dirk Targoni, Robert Bosch GmbH
Carlos Mora-Golding, DENSO Corporation
Dr. Robert Kaster, Robert Bosch LLC
Hank Chavers, Panasonic
Mike Westra, Ford Motor Company
Phil Lapczynski, Renesas Electronics America Inc.
Hassan Ridha, Stellantis
Isaac Snellgrove, Lear Corporation

Report — Editors, Reviewers, and Contributors

Helmut Adler, Vitesco
Kalyan Chakravarthy Ananthoju, Volvo Group
Ben Betcher, Polaris Inc.
Adam Brackmann, Dana Incorporated
Michael Cesarz, Mitsubishi Motors
Hank Chavers, Panasonic
Stephen Christensen, Geotab
Dr. Oliver Creighton, BMW of North America, LLC
Anthony Freilach, Robert Bosch LLC
Charles Hart, Hitachi America, Ltd.
Di Jin, Polaris Inc.
Adam Just, Toyota Motor North America (through April 2024) and West Mountain LLC (present)
Dr. Robert Kaster, Robert Bosch LLC
William Kivett, Cummins Inc.
Phil Lapczynski, Renesas Electronics America Inc.
Roger Marsal, FICOSA Automotive
Matthew Mackay, General Motors
Christoph Menig, Vitesco
Carlos Mora-Golding, DENSO Corporation
Peter Morris, Robert Bosch LLC (retired)
Valentina Penso, Marelli Corp.
Tohyun David Pyun, Sumitomo Electric
Rincy Raju, Dana Incorporated
Hassan Ridha, Stellantis
Isaac Snellgrove, Lear Corporation
Domenic Steinacker, Aptiv
Dirk Targoni, Robert Bosch GmbH
Mike Westra, Ford Motor Company
Patrick Woo, Independent Consultant
Japan Auto-ISAC SBOM Sub-Working Group members

Other Acknowledgements

Many other members and representatives provided valuable feedback and devoted substantial time to helping create, improve, and release this report, including some major contributors who prefer to remain anonymous. The SBOM-WG[2] thanks them sincerely for their contributions.

[1] Automotive Information Sharing and Analysis Center
[2] Auto-ISAC SBOM Working Group

List of Figures

5.1 Declarative Dependency Management 35

9.1 ISO 26262 V-Model (https://about.gitlab.com/solutions/iso-26262/) 60
9.2 Binary Repository as an Archive and Smart Proxy 61
9.3 Vulnerability Information Sources 63

List of Tables

4.1 Software License Types 30
10.1 Languages and Ecosystems 84

Section 1

Introduction

1.1 Industry Background

Automobiles and vehicles are approximately a $3 trillion industry, with products built in more than 50 countries by thousands of companies participating in the supply chain, and ultimately producing more than 60 million vehicles globally per year. The industry and its products require unique consideration of public safety, security, sustainability, and many other areas of policy concern for governments and citizens. As a result, the industry is highly regulated in all markets. In addition, automotive products have become much more complex in the last few years. A modern car can have 30,000 parts and 100 million lines of software and firmware code. The technology can include autonomous decision making, ground and satellite communication, and ultra-short deadline real-time processing on safety-critical Electronic Control Unit (ECU)s. Thus, risks have expanded to include software supply chain integrity and cybersecurity. These areas are now top concerns for automotive industry management globally.

Automotive companies participate in supply chains in one or more of the following roles: Original Equipment Manufacturers (OEMs), Tier-1 suppliers (primarily selling to OEMs), Tier-2 suppliers (primarily selling to Tier-1 suppliers), and Tier-N suppliers, a catchall term for the network of companies who are further upstream in the supply chain. A typical car includes parts, hardware, software, and services from dozens of suppliers.

The major overarching concern of the auto industry and the focus of regulators is the safety of the public including drivers, passengers, other road users, and pedestrians. As a result, the auto industry historically has been a leader in safety innovation. Cybersecurity has been recognized in the past several years as a critical contributor to safety, and that importance is reflected in the newest automotive standards and regulations (e.g. ISO/SAE 21434 [2], UNECE WP.29 / R155 [4], and UNECE WP.29 / R156 [5]) governing cybersecurity practices and risk management.

1.2 SBOM General Background

One of the most basic requirements for ensuring cybersecurity is to thoroughly understand the way a software application works. Virtually all modern automotive software use COTS[1] or OSS[2] to perform one or more functions. Security concerns frequently arise in these underlying software components. In order to understand the security implications of applications that include such components, any risk analyst must know first what components are present in released software.

SBOMs[3] are a means to achieve that goal. An SBOM is a structured, hierarchical list of software libraries and other components that make up a software product. The benefits of such a list are clear: With an understanding of which components are included, risks arising from vulnerabilities can be identified, analyzed and treated by software designers and cybersecurity teams.

Industries broadly agree that SBOMs are critical to managing risk in a complex and increasingly software-driven supply chain. And so, automotive companies are considering how SBOMs can aid vulnerability management for development teams and other supply chain participants.

1.3 Auto-ISAC SBOM Working Group

The SBOM-WG[4] began several years ago to explore the implications of SBOM for the industry. It has grown from its original complement of 9 companies to encompassing the majority of the Auto-ISAC[5]

[1] Commercial Off-the-shelf
[2] Open Source Software
[3] Software Bills of Materials
[4] Auto-ISAC SBOM Working Group
[5] Automotive Information Sharing and Analysis Center

membership. The SBOM-WG published the members-only "Software Bill of Materials Information Report" in 2022 with a focus on formats and requirements. Shortly after publication, the SBOM-WG reconvened to study SBOM operations through practical discussions and exercises.

The SBOM-WG has approximately 50 companies represented. Many of these, including several OEMs and Tier-1 / Tier-2 suppliers, are also recent participants in public SBOM-related events, and the SBOM-WG's findings have substantial overlap with non-confidential projects. However, the project work continues to be classified as member-only TLP:AMBER, and access to its work products is restricted to the Auto-ISAC member companies except when explicitly declassified.

1.4 Automotive SBOM Use

From a compliance and liability point of view, OEMs have responsibility for the entire vehicle including all parts sourced from suppliers. Tier-1 suppliers are responsible to the OEMs for all components of their supplied parts, Tier-2s are responsible to their Tier-1 customers, and so forth. For this reason, it is important to the industry to have an SBOM that can be traced to a particular supplier when a vulnerability is discovered.

As with all large software organizations, it is common practice for auto companies to have tracking systems for open source licenses and contracts. Many companies also have other systems for tracking vulnerabilities in their own products and components. While they may not be compatible with recent SBOM formats, these systems and their data are functionally equivalent in many ways to standardized SBOM systems.

Sharing of SBOMs with downstream supply chain participants is generally favored by the SBOM-WG members and is under consideration by many companies at the time of this writing. However, the auto industry is extremely competitive and any leakage of even small technical details may put companies at a competitive disadvantage. The information in SBOMs is therefore as sensitive and confidential as the software they describe. Therefore, many companies in the auto industry also agree that SBOMs should be shared only with trusted parties. Sharing SBOMs between companies will require proper data protection and access control. Confidentiality between supplier and customer is a major concern for SBOM adopters, and any access to the data should be governed through NDAs[6], CIAs[7], and other legal agreements.

Many companies are optimistic about SBOMs for enhanced vulnerability management, and many also anticipate secondary benefits in related areas like license and IP[8] compliance. Any new technology must be managed appropriately, and SBOMs are no exception. The SBOM-WG exercises and discussions revealed several concerns for SBOM operations, including how to exchange and manage SBOMs with confidentiality and IP protection, limitation of any potential liability, and an orderly and manageable storage and retrieval system.

One high-importance SBOM-WG finding is that automation is essential to SBOM adoption due to the high complexity, vast inventory data, and sheer size of SBOMs. Security tool vendors are actively developing software that will address many of these issues, and in fact there are several available SBOM software tools that are effective in easing adoption and operations.

[6] Non-Disclosure Agreements
[7] Cybersecurity Interface Agreements
[8] Intellectual Property

The auto industry is still at the beginning of SBOM adoption. Process maturity and experience are low at this time for inter-company SBOM exchange, though the industry has some related areas that are well developed.

1.5 Report Objectives and Scope

This Informational Report ("Report") will provide recommendations based on the studies and exercises conducted by the SBOM-WG through 2024. The SBOM-WG recommends that an effective automotive SBOM program should achieve the following outcomes:

- Guide the creation of reliable SBOMs that give supply chain participants visibility through all tiers of the supply chain
- Suggest SBOM-assisted methods for vulnerability management
- Encourage consistent SBOM formats and processes among participants including SMB[9] suppliers
- Guide IP sharing strategies for participants, including mechanisms to ensure confidentiality

The SBOM-WG has published to Auto-ISAC members under TLP:AMBER complementary Reports. The first, published in 2022, primarily covered SBOM formats and construction conventions, and particularly augments this Report's treatment of those SBOM dimensions.

The second was positioned as a guide to practices that the SBOM-WG has exercised or considered that we believe will ease and optimize the transition to widespread use of SBOMs. The third Report, largely based on the second, has been cleared for TLP:CLEAR release for the general public in order to facilitate the common understanding of its contents across the entire industry.

Note *The SBOM-WG's scope of study is limited to vulnerability management and cybersecurity. Software inventory, license management, asset tracking, and other useful operational processes can benefit from SBOMs. However, they are out of scope for this Report.*

1.6 Report Structure and Content

The Report will examine several aspects of SBOMs and will make suggestions for automotive industry practices which can be reasonable options for implementing SBOMs. The sections are as follows:

§1 Background and overview (this section)

§2 SBOM Projects in Government, Industry, and Standards: A short survey of important SBOM and supply chain assurance trends that provide some context for automotive SBOM adoption

§3 Ways the Auto Industry is Unique: Characteristics of supply chains in the auto industry that influence SBOM adoption and practices

§4 Pre-sale Considerations for SBOMs: A discussion of SBOM inclusion during the evaluation and negotiation phases of supplier contracts, e.g. clauses requesting SBOMs, confidentiality and handling of component data, etc.

§5 SBOMs in Development: Some suggestions for building and using SBOM during the design and engineering phases of software and hardware engineering for automotive products

§6 Exchanging and Collaborating with SBOMs Between Supply Chain Participants: Recommendations for effective transmission, receipt, and storage of SBOMs

[9] Small and Midsize Business

§7 Vulnerability Operations Using SBOMs: Options for integration of SBOM techniques for more effective incident and vulnerability response

§8 Review of SBOM Workshop Exercises and Findings: Discussion of the two SBOM-WG live exercises conducted in 2023

§9 Tools Survey and Overview: Automation opportunities for specific processes within overall SBOM operations

§10 Appendices and References

Section 2

SBOM Projects in Government, Industry, and Standards

2.1 Overview

SBOM[1] as a cybersecurity concept is in early stages of adoption worldwide. While several large companies and many open source authors have fully embraced it, much of the automotive industry, is still at the beginning of SBOMs adoption and integration into engineering and vulnerability management. At the time of this writing there are perhaps dozens of SBOM-related and secure software supply chain projects that publish at least some findings or guidance. The following is a brief review of the most relevant.

Note *Some of the examples below do not affect the automotive industry but are included to show the general direction of activity.*

2.2 United States Government

The USG[2] has actively promoted SBOMs since 2018 through public-private partnerships, internal department and agency programs, and EOs[3] and other directives. The automotive industry, while regulated mostly by the DOT[4], is affected by diverse agencies and requirements in less obvious but important ways.

- https://www.whitehouse.gov/wp-content/uploads/2024/02/Final-ONCD-Technical-Report.pdf

2.2.1 Department of Transportation - National Highway Traffic and Safety Administration

The NHTSA[5] is the primary US[6] regulator for the automotive industry and also collaborates with industry groups and companies to improve safety through voluntary techniques and cooperation. NHTSA does not require the use of SBOMs by automotive companies but prominently features SBOMs describing detailed voluntary cybersecurity practices in the September 2022 paper "Cybersecurity Best Practices for the Safety of Modern Vehicles."

- https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/
- https://www.nhtsa.gov/sites/nhtsa.gov/files/2022-09/cybersecurity-best-practices-safety-modern-vehicles-2022-tag.pdf

2.2.2 Cybersecurity and Infrastructure Security Administration

The CISA[7], the cybersecurity arm of the US DHS[8], is one of the most visible agencies of SBOM promotion. They guide all parts of the USG executive branch agencies on cyber best practices. CISA is also the primary oversight body for enforcing presidential EOs and PPDs[9] related to cybersecurity. Well-known among these are:

- **EO**: 14028 "Improving the Nation's Cybersecurity".

  https://www.gsa.gov/technology/it-contract-vehicles-and-purchasing-programs/information-technology-category/it-security/executive-order-14028

[1] Software Bill of Materials
[2] United States Government
[3] Executive Orders
[4] Department of Transportation
[5] National Highway Traffic and Safety Administration
[6] United States of America
[7] Cybersecurity and Infrastructure Security Administration
[8] Department of Homeland Security
[9] Presidential Policy Directives

- **EO**: 13636 “Improving Critical Infrastructure Cybersecurity”.

  https://www.cisa.gov/sites/default/files/2023-01/eo-13636-improving-critical-infrastructure-cybersecurity-508.pdf

- **PPD**: 21 “Critical Infrastructure Security and Resilience”.

  https://www.cisa.gov/resources-tools/resources/presidential-policy-directive-ppd-21-critical-infrastructure-security-and

- **EO**: 13960 “Promoting the Use of Trustworthy Artificial Intelligence in the Federal Government”.

  https://www.whitehouse.gov/briefing-room/presidential-actions/2023/10/30/executive-order-on-the-safe-secure-and-trustworthy-development-and-use-of-artificial-intelligence/

SBOM use figures prominently in these and many other CISA oversight projects.

Since taking over the projects from the DOC[10] National Telecommunications and Information Administration (NTIA) in 2021, CISA has led efforts to codify and promote SBOM and other software transparency techniques. CISA favors industry participation and persuasion over outright regulation to encourage SBOM and other cybersecurity improvements. During this time, CISA has released multiple documents and guidance for commercial entities on SBOM use, including a framework for adoption, FAQs[11], tooling guides, recommended formats, and operational primers. They have also hosted several SBOM-related events, most prominently a periodic public virtual meeting called “SBOM-a-Rama,” featuring industry partners.

CISA has also integrated SBOMs into vulnerability management operations, particularly in the PSIRT[12] guidance. The VEX[13] project aims to formalize vulnerability alerts, allowing for better alert automation and leveraging SBOMs where possible. VEX standardizes and automates software owners’ ability to communicate vulnerabilities and remediation advice to their users, especially PSIRTs.

CISA’s SBOM and VEX projects are now entering their fourth phase since they began in 2018 continuing to assist all software creators, distributors, and consumers with vulnerability management and software transparency.

The Auto-ISAC[14] has influenced and adopted much of CISA’s guidance, with auto industry representatives participating in CISA SBOM program and its predecessors since late 2018. We strongly believe in harmonizing CISA’s guidance with the findings and approaches in this Report.

- https://www.cisa.gov/sbom
- https://www.cisa.gov/sites/default/files/publications/eo-13636-ppd-21-fact-sheet-508.pdf
- https://www.cisa.gov/secure-software-attestation-form

[10] Department of Commerce
[11] Frequently Asked Questions
[12] Product Security Incident Response Team
[13] Vulnerability Exploitability Exchange
[14] Automotive Information Sharing and Analysis Center

2.2.3 Food and Drug Administration

The medical device industry, while very different from the auto industry, shares the core requirement of very high attention to safety and quality. Therefore it is worth comparing and contrasting the industry approaches for regulatory and cybersecurity practices. In the US, the FDA[15] regulates the medical device industry. It was the first regulator to require advanced cybersecurity features in the design, manufacture, and operation of regulated devices. Since 2023, SBOMs are mandatory for FDA approval of any new medical device, marking the first time SBOMs have been mandated by any regulations.

The H-ISAC[16] is building an industry cooperative to enable compliance with this regulation (see below for more information).

- https://www.fda.gov/media/173516/download?attachment
- https://www.fda.gov/medical-devices/digital-health-center-excellence/cybersecurity-medical-devices-frequently-asked-questions-faqs

2.2.4 Department of Defense

The US DoD[17] is one of the largest customers for many industries globally and is a major buyer of automotive products including cars, trucks, military vehicles, and related software and equipment. DoD has an extensive requirement list for both military and non-military equipment, including the FAR[18], DFARS[19], and the FedRAMP[20] which is primarily targeted at cloud services.

It has also recently highlighted the need for SBOMs as a key component of cybersecurity supply chain risk management. There is no current regulation requiring SBOMs but a proposal to add it as a requirement in FAR has completed the public comment phase (as of December 4, 2023) and appears likely to be adopted in some form.

Based on these trends, automotive industry vendors to DoD and related departments should be prepared for SBOMs to be required for software products. While SBOMs for vehicles and other cyber-physical systems have not been explicitly covered, they are also likely to be required by a growing number of DoD acquisitions.

- https://www.acquisition.gov/browse/index/far
- https://www.acquisition.gov/dfars
- https://www.fedramp.gov/program-basics/

2.2.5 National Institute of Standards and Technology

The National Institute of Standards and Technology (NIST) is part of DOC. As the primary standards organization in the US, NIST has focused on cybersecurity for the last several years. Directed by the White House, NIST publishes standards for SBOMs and other software and hardware cybersecurity features. Although not a regulator, NIST's actions affect the automotive industry by defining specifications for regulations and requirements issued by other USG agencies.

- https://www.nist.gov/itl/executive-order-14028-improving-nations-cybersecurity/software-security-supply-chains-software-1

[15] Food and Drug Administration
[16] Health Information Sharing and Analysis Center
[17] Department of Defense
[18] Federal Acquisition Regulation
[19] Defense Federal Acquisition Regulation Supplement
[20] Federal Risk and Authorization Management Program

2.2.6 Department of Energy

The US DoE[21] is primarily a regulator of the electric power industry as well as some energy supply chain systems such as fuel and chemical pipelines. It also is a regulator for the nuclear industry and oversees both generating nuclear fuel and nuclear weapons operations. DoE's purview has obvious national security implications, and the agency has a restrictive information sharing policy outside the industry.

DoE manages several National Laboratories, many of which have active public-private cybersecurity programs which are partially open to the public and other non-energy industries. The INL[22] is actively engaged in creating and promulgating SBOM guidance for the energy industry and runs a public SBOM PoC[23].

Certain parts of DoE guidance are directly applicable to the auto industry, particularly its authority over the the US power grid, which includes EV[24] charging stations. Other DoE programs serve as examples of cybersecurity trends in critical infrastructure.

- https://sbom.inl.gov/
- https://www.energy.gov/ceser/articles/ceser-partners-cisa-release-new-framework-software-bill-materials-sharing-0

2.2.7 Federal Communications Commission

The FCC[25] administers the Cyber Trust Mark, a label on IoT[26] equipment certifying compliance with a minimum set of security and supply chain requirements. Note: The program excludes motor vehicles and motor vehicle equipment because they are regulated by NHTSA.

- https://www.fcc.gov/document/fcc-adopts-rules-iot-cybersecurity-labeling-program

2.3 Governments - European Union

2.3.1 Cybersecurity Resilience Act and Related Regulations and Guidance

The EU[27] passed the CRA[28] provisionally in early 2024 and it is scheduled to become law in early 2027. It requires SBOMs for a wide range of products. Vehicles and certain other products are regulated by other rules and are currently exempt from CRA.

- https://digital-strategy.ec.europa.eu/en/policies/cyber-resilience-act

2.3.2 Federal Office for Information Security - Germany

Germany's government agency for cybersecurity, the BSI[29], is considering SBOMs as part of CRA. It is unclear whether any guidelines will apply to the automotive industry, but it appears likely that BSI will defer in the same way CRA has.

[21] Department of Energy
[22] Idaho National Labs
[23] Proof of Concept
[24] Electric Vehicle
[25] Federal Communications Commission
[26] Internet of Things
[27] European Union
[28] Cybersecurity Resilience Act - European Union
[29] Federal Office for Information Security - Germany

Notably, BSI has developed a vulnerability notification standard called the CSAF[30]. The US CISA and private working groups have adopted CSAF as a leading format for VEX program, where suppliers can advise vendors on vulnerability exploitability, and provide remediation activities. SBOMs and CSAF, along with vulnerability data from the National Vulnerability Database (NVD), allow end users to automate vulnerability processing, first determining if a vulnerable component presence in the software inventory, then acting according to supplier guidance.

- https://www.bsi.bund.de/EN/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Technische-Richtlinien/TR-nach-Thema-sortiert/tr-nach-thema-sortiert_node.html
- https://owasp.org/www-chapter-frankfurt/assets/slides/58_OWASP_Frankfurt_Stammtisch_1.pdf

2.4 Governments - Japan

2.4.1 Ministry of Economy, Trade and Industry

The Japanese government has become keenly interested in SBOM after a long period of study to understand the direction of global programs, and it has begun to consider active guidance. The METI[31] released a paper in 2023 that specified an SBOM program and guidance similar to the US CISA's, and the Japanese automotive industry is now studying implementation of SBOMs in cooperation with METI and other industry groups.

- https://www.meti.go.jp/english/press/2023/0728_001.html

2.5 Industry

2.5.1 Health Information Sharing and Analysis Center

H-ISAC has participated in a long-running PoC for SBOM utilization between the MDMA[32] members and HDOs[33] (e.g. hospitals and clinics).

H-ISAC also began a pilot SBOM clearinghouse in which the MDMA members can upload their SBOMs to a central, access-controlled database which is accessible to the US FDA for compliance, and selectively accessible to customers HDOs.

- https://health-isac.org/tag/sbom/

2.5.2 Energy - Public/Private Proof of Concept (Sponsored by US Department of Energy Idaho National Labs

DoE is mandated by the President to ensure cybersecurity in the critical infrastructure sector of energy generation and delivery. Among its initiatives is a long-running public SBOM PoC program that brings energy generation and grid equipment suppliers together with electric utilities to understand and develop SBOM techniques. Note: There are several other programs from DoE and by industry that are restricted to industry participants.

- https://sbom.inl.gov/

[30] Common Security Advisory Framework
[31] Ministry of Economy, Trade and Industry - Japan
[32] Medical Device Manufacturers Association
[33] Health Delivery Organizations

2.5.3 Information and Communications Technology

The ICT[34] industry likely has the deepest and broadest range of SBOM programs, driven by government guidance, regulation, and the volume of ongoing security breaches. Here are some notable examples:

- Red Hat has a comprehensive security information program including newer implementations of SBOM and VEX and some informative process diagrams.

  https://www.redhat.com/en/blog/future-red-hat-security-data

- Cisco has been involved in SBOM programs since at least 2018 with the US NTIA and CISA.

  https://sec.cloudapps.cisco.com/security/center/resources/sbom-faq

- Google's Android project has extensive developer pages providing step-by-step instructions for SBOM creation. In addition, many other Google projects touch on SBOM and related security approaches.

  https://source.android.com/docs/setup/create/create-sbom

2.6 Academia, Standards, and Open Source

2.6.1 Society of Automotive Engineers

In January 2024, the SAE[35] became the first standards body to consider automotive industry SBOMs. The effort is a sub-task-force under the Vehicle Cybersecurity Systems Engineering Committee (TEVEES18A) and is cooperating actively with the Auto-ISAC's SBOM projects to harmonize standards and practices.

SAE also cooperates with the ISO[36] on multiple standards for the automotive and aviation industries. The most relevant of these is ISO/SAE 21434 [2]. SAE will continue to collaborate with ISO on new versions of applicable standards, and new versions of ISO/SAE 21434 [2] may include SBOMs.

- https://standardsworks.sae.org/standards-committees/vehicle-cybersecurity-systems-engineering-committee

2.6.2 International Organization for Standardization (ISO/SAE 21434, 5962:2021, TC 292/22373, 26262)

ISO is active in setting standards for many industries, including automotive. Several of its standards relate to SBOMs, including ISO/SAE 21434 [2], 5962:2021 SPDX 2.2.1, TC292/ISO 22373 Cybersecurity and Resiliency in Supply Chains, and ISO 26262 [1]. ISO has not yet promulgated a standard for SBOMs in any industry, but the topic is under consideration. The EU Parliament's CRA SBOM requirement is likely to accelerate inclusion of SBOMs in several ISO standards.

- https://www.iso.org/standard/70918.html
- https://www.iso.org/standard/81870.html
- https://www.iso.org/committee/5259148.html
- https://www.iso.org/standard/68383.html

[34] Information and Communications Technology
[35] Society of Automotive Engineers
[36] International Organization for Standardization

2.6.3 UNECE Regulations

The UNECE[37] WP.29[38] has ratified two rules closely related to SBOMs. UNECE WP.29 / R155 [4] specifies that the Original Equipment Manufacturers (OEMs) must monitor and track cybersecurity issues with their vehicles. UNECE WP.29 / R156 [5] requires an update mechanism to load new software onto the vehicle to patch vulnerabilities.

UNECE WP.29 / R155 [4] suggests that SBOMs are necessary for managing cybersecurity issues on a vehicle. Though the rule does not explicitly refer to SBOMs, it demands risk assessments, supplier risk monitoring and remediation, and vulnerability/incident management for supplier-induced cybersecurity problems. These are all requirements for the cybersecurity management system which is the main focus of the regulation.

- https://unece.org/transport/documents/2021/03/standards/un-regulation-no-155-cyber-security-and-cyber-security
- https://unece.org/transport/documents/2021/03/standards/un-regulation-no-156-software-update-and-software-update

2.6.4 Open Source Security Foundation - Linux Foundation

Both the LF[39] and its sponsored project the OSSF[40] have built strong SBOM programs in several areas. LF, among other accomplishments, is the originator of the SPDX[41], a full-featured descriptor format for software composition and related data. SPDX was originally built as a tool for open source license management and tracking, and has a robust license notation syntax. However, SBOMs are a natural extension of component license tracking and SPDX is well- suited for SBOMs as well.

SPDX, along with the OWASP[42] CycloneDX (see below), is one of two formats recommended by the US DHS CISA and other SBOM proponents for creation and operation of SBOMs. Additionally, ISO released SPDX as standard: ISO/IEC 5962:2021 Information technology - SPDX® Specification V2.2.1.

- https://spdx.dev

OSSF sponsors a wide range of public discussions and projects including a substantial number that directly address SBOM issues. OSSF maintains an overall focus on practice recommendations and tooling in the open source world.

- https://openssf.org

2.6.5 Open Worldwide Application Security Project

Like LF, OWASP has an ambitious SBOM program with multiple tools, standardized formats, and a developing ecosystem for adopters. Its main SBOM work is part of the CycloneDX project, which specifies a format that can be used for several different software supply chain functions including vulnerability alerting for downstream consumers and customers, inventory management (especially for SBOM components), and vulnerability operations.

[37] United Nations Economic Commission for Europe
[38] World Forum for Harmonization of Vehicle Regulations
[39] Linux Foundation
[40] Open Source Security Foundation - Linux Foundation
[41] Software Package Data Exchange
[42] Open Worldwide Application Security Project

CycloneDX is a popular format for SBOM exchange due to the wide range of functions it supports in software supply chain risk management. CycloneDX along with SPDX is recommended by CISA and NIST as an SBOM format, and is one of three emerging next-generation vulnerability notification formats (along with CSAF and VEX). OWASP website offers a range of tools, including scanners, format converters, vulnerability analyzers, and other utilities.

- https://owasp.org/www-project-cyclonedx/

2.6.6 Organization for the Advancement of Structured Information Standards

The Organization for the Advancement of Structured Information Standards (OASIS) has a number of projects that are targeted for improving software supply chain risk management. The most prominent example for the automotive industry is the OSIM[43]. The OSIM is a project to mature and extend SBOMs and other supply chain component data into a comprehensive data model. The vision for the OSIM is to allow traceability of both SBOM and other technical attributes as well as data on physical components.

- https://www.oasis-open.org/2024/05/02/introducing-oasis-open-supply-chain-information-modeling/

2.6.7 GitHub

SBOM operations require a wide variety of tools throughout the software lifecycle. The well-known repository librarian GitHub has embraced SBOM operations with a series of enhancements to repository and component metadata that ease the distribution of SBOMs and enhance software identification, license management, and vulnerability operations. Both GitHub-native and third party enhancements are available. These include tools for central SBOM storage (OmniBOR/gitbom), standardized conventions for software component identification (git-hash-object), including SBOMs in GitHub packages (Dependency Graph), and many others.

- https://github.com
- https://github.com/OmniBor
- https://docs.github.com/en/code-security/supply-chain-security/understanding-your-software-supply-chain/exporting-a-software-bill-of-materials-for-your-repository
- https://git-scm.com/docs/git-hash-object

[43] Open Supply Chain Information Modeling

Section 3

Ways the Auto Industry is Unique

3.1 Introduction

In the past, vehicles were seen as mechanical transportation devices. Passenger cars, trucks and motorcycles moved people and things mechanically, but vehicle technology had not advanced enough to elevate related security concerns beyond a secondary role. Internal communication networks were unprotected as there was no real communication outside of the closed system within the vehicle.

The automotive industry is rapidly integrating advanced software and internet connectivity, creating new features which allow the use of a wide range of software, connectivity, and cloud services. This creates new attack surfaces and unique security challenges. Today's vehicle can be characterized as a collection of highly specified Electronic Control Unit (ECU)s, uniquely entrusted with all associated functionalities. For example, the ABS[1] ECU controls the braking system. All functions linked to braking the vehicle find their home in this ECU. Furthermore, development of autonomous vehicles extends the control challenge and creates fundamentally new cybersecurity concerns. Contrary to some popular metaphors, vehicles are not computers or smartphones on wheels. They are cyber-physical systems that use sophisticated computer processing to safely control a large and complex machine, with limited on-board resources and demanding real-time requirements. Thus automotive development, operations, and update processes are highly regulated. These factors make security for vehicles a major and growing concern for the automotive industry and the public.

The automotive industry faces specific challenges for security that are further explored in the following subsections.

3.1.1 Converging Systems

Traditional mechanical components are controlled by numerous ECUs equipped with chips with limited resources, and complex software. These ECUs are increasingly connected to the internet. This connectivity increases vehicle functionality and offers various conveniences to users, but it also opens numerous ways for cyber threats. Security incidents can impact physical safety, IP[2], and data privacy.

3.1.2 Long Lifespan

Vehicles are expected to last beyond 15 years, making them vulnerable to security exploits that emerge over time. Patching vulnerabilities in millions of geographically dispersed vehicles poses significant challenges. The challenges include development, approval, compatibility with old hardware technologies, limited resources, logistics and distribution, and preservation of asset lists and know-how.

3.2 Complex Supply Chain

The automotive industry boasts one of the most intricate and globalized supply chains in the history of the world. Modern vehicles have numerous ECUs, components, parts, and software which contain millions of lines of code and are sourced from various suppliers worldwide, creating multiple points of vulnerability for potential exploitation. Each supplier represents a potential entry point for cyberattacks or malicious interference. Vulnerability monitoring against available open-source databases like the National Institute of Standards and Technology (NIST)'s National Vulnerability Database (NVD) has limitations. Not all components will or can be present in such databases. Therefore, all supply chain participants need to be involved in vulnerability monitoring, analysis and maintenance activities. Ensuring consistent security standards across this complex structure is challenging and requires the

[1] Anti-lock Braking System
[2] Intellectual Property

involvement of the Original Equipment Manufacturer (OEM) and all tier levels for vulnerability monitoring, analysis, patch development, approval, testing and deployment of security patches and respective infrastructure.

3.3 Cybersecurity and Safety

In the automotive industry, safety-critical systems must be fault-tolerant (resilient to hardware, software, and mechanical failures) to prevent accidents and maximize occupant safety. Functional safety design principles and verification for safety-critical systems indicate rigorous standards and comprehensive testing and validation procedures.

Safety and security are inherently linked in the automotive industry, and design teams are fully aware that there can be no safety without security. Safety requirements focus on protecting occupants, pedestrians, and other road users from physical harm due to vehicle accidents or malfunctions. In the same way, security measures must prevent unauthorized access, manipulation, or disruption of critical services which could compromise safety functions like braking or steering. An integrated approach considers both safety and security aspects to ensure comprehensive protection for vehicles, drivers, passengers, and the public. This approach requires a focus on secure software development practices and ongoing collaboration throughout the industry to stay ahead of evolving threats.

3.4 Updates and Complexity

Updating or upgrading vehicle software requires precise planning and testing to ensure compatibility, stability, and functionality. The complexity of interconnected systems with long life cycles amplifies the challenges associated with updates and upgrades. Even as modern vehicles become more like smartphones, they cannot be patched in the same way due to the need for rigorous safety and functionality testing.

3.5 Risk Assessment and Vulnerability Disclosure

To fulfill safety requirements, standards and good practice require thorough risk assessments to identify potential hazards, mitigate risks, and prevent undetected failures. Similarly, security risk assessments require detailed reviews of vulnerabilities, threats, and potential consequences of security breaches. By conducting holistic risk assessments that encompass both safety and security considerations, automotive manufacturers can develop robust strategies to minimize risks and enhance overall vehicle safety and security.

Similar requirements drive vulnerability disclosure. Security considerations are increasingly integrated into functional safety standards to address emerging cyber threats and ensure the resilience of safety-critical systems against malicious attacks. Responses to security measures must bring the vehicle into a secure state without jeopardizing safety goals. Therefore, security disclosure planning must follow a predefined process and factor the impact of both the vulnerability and the disclosure on vehicles and stakeholders. An uncontrolled or improper disclosure may unintentionally put the social and economic welfare of customers at risk.

3.6 Cost and Resources Limitation

As in all industries, automotive companies are profit-driven businesses that have a strong focus on cost. To remain economically viable, suppliers must include security costs in the pricing of parts and services for customers at all tiers of the supply chain. The need for security updates to technology,

combined with the long service life of products, adds significant future costs and complicates cost computation.

3.7 Data Privacy Concerns

Vehicles are becoming data-rich environments. Safety and other market requirements often involve collecting, processing, and transmitting large amounts of potentially sensitive information about drivers, passengers, vehicle operations, and environmental conditions. This data may include personal details, location history, driving patterns, and even biometric data in advanced systems. The automotive industry must consider how to protect this data from unauthorized access, tampering, theft, or misuse to safeguard privacy and prevent potential harms such as identity theft or surveillance. Compliance with data privacy regulations, such as the GDPR[3] or the CCPA[4], ensures that safety-related data is handled securely and responsibly, minimizing the risk of privacy breaches or data misuse.

3.8 Autonomous and Semi-Autonomous Technologies

The development and introduction of autonomous and semi-autonomous vehicles introduces a new dimension of security challenges for the automotive industry. These vehicles receive input information from advanced sensors, cameras, radar, LIDAR[5], and use sophisticated algorithms to perceive environment, make decisions and navigate the environment. Any compromise to these systems could lead to malicious manipulation of vehicle behavior and possible serious accidents, posing significant safety risks.

3.9 Regulatory Compliance

The automotive industry operates within a stringent regulatory framework aimed at ensuring vehicle safety, environmental standards, and data privacy on national and international levels. Regulations and standards govern all phases of the product lifecycle, including development, updates, and decommissioning. Compliance with these regulations imposes additional security requirements on manufacturers and suppliers, ranging from cybersecurity protocols to data protection measures. Failure to adhere to these standards can result in legal consequences, financial penalties, and reputational damage.

For example, adherence to safety regulations and standards such as ISO 26262 [1], which specify requirements for achieving functional safety in electronic systems within vehicles, is mandatory for automotive manufacturers. Similarly, compliance with cybersecurity regulations and industry standards, such as UNECE WP.29 / R155 [4] or ISO/SAE 21434 [2], is essential for addressing cybersecurity risks and demonstrating a commitment to protecting vehicles against cyber threats. These standards outline processes for identifying and mitigating hazards associated with system malfunctions or failures. Regulatory compliance frameworks provide guidelines and best practices for implementing effective safety and security measures, fostering trust among consumers and regulatory authorities.

3.10 Physical Security Threats

Beyond cyber threats, the automotive industry faces traditional physical security risks such as theft, vandalism, and sabotage. Manufacturing facilities, distribution networks, and vehicle storage areas are susceptible to intrusions or attacks, potentially disrupting production, causing financial losses, or compromising vehicle integrity.

[3] General Data Protection Regulation - European Union
[4] California Consumer Privacy Act
[5] Light Detection and Ranging

3.11 Interconnected Infrastructure

The automotive ecosystem extends beyond individual vehicles to encompass broader infrastructure such as traffic management systems, other vehicles in near range, charging stations for electric vehicles, cloud services, automated parking environments and smart transportation networks. These interconnected systems create interdependencies and vulnerabilities, where a security breach in one area could cascade into widespread disruptions or safety hazards.

3.12 Secure-by-Design Principles

Safety-critical systems in vehicles are designed following secure-by-design principles to mitigate risks associated with software vulnerabilities, design flaws, or system failures. Incorporating security features and countermeasures as early in the design and development process as possible helps prevent security vulnerabilities from compromising the safety and reliability of automotive systems. Secure coding practices, threat modeling, static code analysis and penetration testing are only a few examples to ensuring that safety-critical systems remain resilient to cyber threats throughout the whole lifecycle.

3.13 Conclusion

Overall, safety requirements exert a significant influence on security considerations within the automotive industry. They shape the design, development, and deployment of vehicles with a focus on protecting occupants, data, and critical systems from harm or exploitation. Robust cybersecurity fortifies functional safety and aligns the quality interests of all stakeholders. By adopting a holistic approach that integrates safety and security principles, automotive manufacturers can enhance vehicle safety, resilience, and trustworthiness in an increasingly connected and digitized automotive ecosystem. In conclusion, the automotive industry's unique blend of technological innovation, global supply chains, data-driven operations, and regulatory requirements, limitations in power consumption, costs, and computing power underscores the critical importance of prioritizing security at every level.

Section 4

Pre-sale Considerations for SBOMs

4.1 Introduction

Note *The contents of this section rely on the concepts defined in ISO/SAE 21434 [2].*

The pre-sale and contract negotiation phase is an opportunity for the client and the supplier to come to agreement on specific policies, infrastructure, and elements that will shape the nature of the product. During this period, there are typically a number of formal documents exchanged, including the RFP[1], detailed specifications including those for cybersecurity, along with the CIA[2] and the DIA[3] that will govern design and production for the successful bidder.

SBOMs[4] are a key artifact that will be used to synchronize the parties on the content and in some ways status of development. Just as importantly, SBOMs and related vulnerability disclosures will be an important ingredient helping to coordinate vulnerability management between supplier and client. Finally, given confidentiality considerations, any contract and pre-sale agreement will need to set mutual expectations for the exchange and management of SBOMs if they are being used.

This section reviews some potentially relevant ideas and actions that will help in adopting SBOMs across the automotive supply chain. Note as with all other contract and pre-sale requirements, these actions and ideas will vary widely in normal business depending upon the supplier and customer. These are in no way presented as best practices or recommendations. Rather, they should be considered as concepts for the awareness of suppliers and customers.

4.2 Request for Proposal and Request for Quotation

Both the RFP and the RFQ[5] stages of purchase negotiation involve exchange of potential proprietary information and usually contain at least a minimal requirement for data confidentiality. While this is not industry practice today, if SBOMs are required to respond to either an RFP or RFQ, it is reasonable to expect SBOMs to be protected also. One way to ensure this protection is in place is to create a specific agreement clause for SBOMs. Another is to adopt a pre-sale CIA (see below). It is possible also that other confidentiality agreements will apply during the RFP or the RFQ process, and automatically apply to SBOMs if they are required or provided.

Note that ISO/SAE 21434 [2] RQ-07-03 requires that any RFQ include the specification that the product will conform to the standard.

4.3 Non-Disclosure Agreement

Non-disclosure of SBOM data has been a focus of the SBOM-WG[6] members since the beginning of SBOM discussions. Confidentiality is one area where the automotive industry has diverged from other SBOM efforts. The industry, as represented by the SBOM-WG, strongly believes that SBOM data is proprietary and must be managed and secured as confidential information from all parties who do not specifically need the information for management of software development, maintenance, vulnerability, and incident management.

[1] Request for Proposal
[2] Cybersecurity Interface Agreement
[3] Development Interface Agreement
[4] Software Bills of Materials
[5] Request for Quotation
[6] Auto-ISAC SBOM Working Group

This Report strongly urges the parties to include SBOMs in their formal agreements. The nature of SBOMs and the information they contain highlights that careful coverage of SBOM topics in formal agreements protects the parties and ensures the SBOM information is used as intended. Ideally, SBOM data should be shared with all downstream customers up to the Original Equipment Manufacturer (OEM) but not with end retail customers for either vehicles or parts. While the SBOM-WG recommends strong confidentiality measures, exceptions and custom agreements are in no way excluded if the parties agree to expanded access to SBOMs.

It is reasonable to include definitions of SBOMs in any document where they are referenced, and the NDA[7] will in many cases be the first definition of SBOM information and its confidentiality. Definitions of an SBOM in agreements can include, among others:

1. Definition of sensitive materials in the SBOM
2. Exemptions to the NDA for any appropriate aspects of SBOM data
3. Permitted disclosure of SBOM to affiliate parties (potentially including internal company access restrictions)
4. Required disclosure of SBOM to government authorities

NDAs are useful for technology knowledge and IP[8] transfer, and many companies have stock NDA frameworks. It is prudent to include:

1. Obligations required of all parties to the NDA
2. Date ranges when the NDA is effective
3. Governing law e.g. state, province, country, etc.
4. Ownership and Confidential information e.g. AUTOSAR[9] BSW[10] owned stack

The NDA will typically provide the framework to protect the SBOM information from being disclosed to unauthorized parties, plus specify the information that is considered sensitive, and how to handle it. SBOM-specific requirements can be included in a general NDA, or in an NDA that is targeted more narrowly to the SBOM.

Product development teams will be in a good position to provide input to the purchasing and legal teams when drafting NDAs for SBOMs.

4.4 Cybersecurity Interface Agreement and Development Interface Agreement

CIAs and DIAs are formal agreements that define management, data exchange, and responsibilities for design and development of products that are relevant to cybersecurity and development. They are required by ISO/SAE 21434 [2] but the standard does not provide a detailed definition of document content beyond general requirements. As a result, there are a range of acceptable terms and structures for CIAs and DIAs.

One primary purpose for interface agreements is to clarify the people and functions who are participating in any cybersecurity development activities. The list of functions, people, and roles is known by the acronym RASIC[11]. RASIC on the products and applications is generally defined in CIAs and DIAs in accordance with ISO/SAE 21434 [2] guidelines. The CIA and the DIA are appropriate documents

[7] Non-Disclosure Agreement
[8] Intellectual Property
[9] AUTomotive Open System ARchitecture
[10] Basic SoftWare
[11] Responsible, Accountable, Supporting, Informed, Consulted

to outline the responsibilities of design, development, testing, monitoring, etc. and they may include SBOM as a routine security activity. There is a distinct advantage to using CIAs and DIAs for SBOM requirements, because SBOMs will be folded into regular security development operations like any other.

In addition, interface agreements specify which data should be exchanged and accessible to which individuals and roles. This too applies to SBOMs, especially if any suppliers require particular confidentiality and protection provisions.

Activities governed by CIAs and DIAs are known as Work Products [2]. One recommendation for SBOM agreements between suppliers and customers is to integrate SBOM in existing ISO/SAE 21434 [2] Work Products [2]. Some suggested Work Products [2] to which SBOMs may be included follow here:

1. Continuous cybersecurity activities
    a) Cybersecurity monitoring: (WP-08-01,02,03)
    b) Vulnerability analysis and management: (WP-08-05,06)
2. Concept Phase
    a) Item definition (WP-09-01)
3. Product Development Phase
    a) Cybersecurity requirements for post-development:(WP-10-03)
    b) Component and application penetration test: (WP-10-04,05,06,07)
    c) Vulnerability scanning and analysis and report: (WP-10-04,05,06,07)
4. Post-Development Phases
    a) Cybersecurity incident response plan: (WP-13-01)

4.5 Identity and Access Management

In many cases, suppliers and customers may not require explicit access control and may trust the partner to ensure appropriate access. However, if one party to an agreement is particularly concerned with restricting SBOM access to certain named individuals or roles within the other party's organization, these access controls should be specified in an appropriate agreement (see §9.5 Identity and Access Management and §B.5 Identity and Access Management). For access granted to individuals, include the individual's name, role, access duration, review/audit dates, and technical methods for restricting access. While a custom document may be created, it may be easier to include any IAM[12] parameters in other documents such as the CIA.

4.6 Know Your Suppliers

A wide range of factors drive supply chain governance, from quality management to geopolitical risk and regulation. Therefore, it is increasingly important for supply chain participants to carefully evaluate their direct and indirect suppliers at all levels of a software supply chain. The supply chain for software in particular encompasses a wide range of dependencies on organizations and individuals all of whom exert influence and provide input into the final software for a product. When new dependencies are identified and added to the software configuration, those dependencies, and all their dependencies

[12] Identity and Access Management

transitively convey to the software product(s). Beyond the software product's composition and components, supply chain participants have privileged access to processes, systems, organizations and people that outside members lack.

4.7 Technology Infrastructure and the Mechanics of Integration

Increased technology integration can often increase productivity within supply chains by automating key activities during the collaboration between contract parties. The contract phase is a good time to explore organizations' technology suppliers and providers relevant to software product collaboration efforts.

SBOMs nearly always require automation using software tools during development, update, and support. It may be advantageous to agree to use the same or similar software tools during each lifecycle phase to ensure compatibility and suitability. Examples of specific infrastructures and tools are discussed in §9.

4.8 Cryptography Algorithms and Providers

SBOMs should be authenticated and integrity checked by digital signature verification. In the absence of digital signatures, SBOMs can be integrity checked by cryptographic hash verification. SBOMs additionally provide integrity of components by cryptographic hash for the component references. A complete and detailed SBOM includes its own digital signature, and cryptographic hashes for any included file or component. Multiple hash algorithms per component are supported, and advanced integrity checking techniques can include their collective verification (see §9.11 Cryptography Software and §B.6 Cryptography Software).

The contract phase gives the parties an opportunity to align on key cryptographic elements, including preferred hash algorithms, and cryptography providers. The importance of the correctness and security of cryptographic implementations cannot be overstated. Defense in depth strategies may require the use of multiple providers. In the US[13], the National Institute of Standards and Technology (NIST) is an important source for up-to-date guidance on cryptographic algorithms and providers. Therefore, selections should align with NIST recommendations and best-practices.

4.9 Confidentiality and Intellectual Property

As outlined in §4.3, confidentiality between parties is essential to the contract process. It is understood the organizations have not only IP, but also trade secrets that must be protected. The disclosure of trade secrets may cause severe financial impact to an organization, and all parties should assiduously seek to avoid it. However, between supply chain participants, SBOMs should include the correct level of detail, in particular on software composition and provenance, that allows them to meet any expected requirements. One of the challenges in automotive product development is finding the correct balance between these competing interests.

4.10 Incident Management and Vulnerability Disclosure

The contract phase is an opportune time to address incident management and vulnerability disclosure. Incident management teams will need to coordinate between supply chain members. Vulnerabilities discovered during any phase of the product lifecycle are delegated to the responsible supplier to remediate. Like updates not targeting vulnerability fixes, the authority to release an update to a vehicle

[13] United States of America

in the field rests with the OEM. When outsiders like researchers or developers find vulnerabilities, they may use an existing coordinated vulnerability disclosure process to alert affected organizations about their findings, and to give them a chance to address them. However, they may not attempt or be able to find the responsible parties or use a coordinated disclosure process as intended. Some programs offer "bug-bounties" for responsible coordinated disclosure. In the worst case, a supply chain participant may learn of a zero-day (0-day) vulnerability in the software product.

Rapid diagnosis and resolution of these issues are fundamental reasons for SBOMs. Vulnerabilities are often found in underlying software libraries and components, and for that reason these supplier/-customer agreements may explicitly include disclosure and management processes using SBOMs for diagnosis and remediation. Supply chain participants may suspend SBOM redactions and confidentiality clauses by mutual agreement for incident management.

4.11 Service Level Agreements

The agreement to work together in the contract phase should be shaped by roles, responsibilities, deliverables, and timelines agreed between the parties. Just as delivery milestones outline key deliveries and dates expected, SLAs[14] document expectations on responsiveness for the contract parties. It is impossible to know in advance exactly when certain events occur. So, an organization needs to understand their own processes, and those of their suppliers, to be able to set expectations appropriately with other contract parties. These SLAs may obligate the parties during the engagement on matters such as:

- The time duration after a vulnerability is discovered or disclosed to a contract party before they notify the other party
- The time duration after a software defect is discovered before it is triaged by the supplier
- The time duration after a minor defect (e.g., documentation defect) is discovered before it will be addressed by the supplier
- The time duration for support request resolution

SBOMs should be included in any SLA requirements in any pre-sale paperwork since they will affect the workload for the supplier and the post-sale security plans for both supplier and customer. Possible SLA requirements could include timing of SBOM delivery (before, during, or after delivery of the SBOM-modeled software product), maximum interval for delivery of SBOMs, certification of receipt, transmission methods, and secure management of SBOM data for both parties.

4.12 Licensing

While the focus of this Report is the use of SBOMs for cybersecurity, they are also a powerful tool for managing software licenses. Licensing is important in pre-sale planning because IP mistreatment can cause significant financial damage to the owner, and financial liability for the IP violator. In extreme cases just one license violation can negate all other IP rights in a software product. Supplier and customer contracts should explicitly state management expectations and liabilities.

Agreements should specify not only the license(s) of the developed product, but also the acceptable and unacceptable licenses or license-styles for the developed product and all its dependencies. SBOMs shared among supply chain participants can transparently identify software licenses for all components of the supplied software product. When good SBOMs are provided, license management can be automated at all levels of the supply chain. A summary of licenses often encountered in automotive software can be found in Table 4.1.

[14] Service Level Agreements

Table 4.1 Software License Types

License Type	Also Known As	Description	Examples
Proprietary	Commercial	Software is typically sold by a software vendor or distributor with a custom license contract	Secure boot libraries for ECUs; TCP/IP and Bluetooth firmware; VXworks, QNX
Consortium	-	Software license granted to members of group	AUTOSAR, FlexRay
Public Domain	Freeware	Free software with rights disclaimed	Adobe PDF Reader; VLC Media Player
Copyleft	Viral license	Free software with rights retained and obligations conveyed. May have few or greater restrictions on use spelled out in custom license terms	Mozilla products (weak IP protection), Watcom, Drupal, MySQL (strong IP protection)
Permissive	Academic license	Free software with permissive use / distribution / relicensing	Apache, BSD

Open-source software used in automotive can exhibit complex licensing because of the nature of transitive dependencies. Open-source software is often classified as “copyleft” or “permissive.” Software components can also be governed by multiple licenses. Also, complex and customized licenses can govern individual software components and are especially common for commercial software. Finally, firmware needed to operate hardware is often proprietary.

Software component SBOMs may indicate different licenses or license expressions. For example, in the SPDX[15] specification, the “declared” license is the one that the software component authors believe govern the use. The “concluded” license is the license or license expression that the SBOM author(s) believe govern the component. There are additional facilities to explain discrepancies.

Copyleft open-source software and proprietary software licenses often convey license obligations as part of the acceptable use. For the subject software product to be compliant with the license, those obligations must be met. SBOM license transparency is a key enabler for validating license obligations and thus a compliant software product.

Even with notional agreements between parties, some component licenses may need to be held in abeyance until the contract parties can legally evaluate the licenses and classify them as acceptable or unacceptable.

[15]Software Package Data Exchange

4.13 SBOM Formats

The SBOM-WG recommends that automotive supply chain participants use one of the US NIST endorsed formats, i.e. either SPDX or CycloneDX SBOM formats in JSON[16]. At the time of this writing, SPDX **v3.0** and CycloneDX **v1.6** are current (see §B.1 CycloneDX Tools and §B.2 Software Package Data Exchange Tools).

In general, both SPDX and CycloneDX have similar functionality, and both can be used for SBOMs effectively. However, they do require conversion to interoperate, and as an additional complication either JSON or a native syntax for each format can be used. For that reason, it may be prudent to specify which standard and notation should be expected by both parties in a supplier/customer agreement.

4.14 Vulnerability Exchange and Disclosure Formats

Using well-known machine-readable formats can be highly effective in communicating vulnerability information. The Report recommends that contracts specify the CSAF[17] format for vulnerability disclosure and using the VEX[18] profile in CSAF for ongoing exploitability and remediation information.

The SBOM-WG further recommends that contracts, at a minimum, specify the VDR[19] for vulnerability disclosure.

- https://www.cisa.gov/sites/default/files/2023-04/minimum-requirements-for-vex-508c.pdf
- https://oasis-open.github.io/csaf-documentation/

[16] JavaScript Object Notation
[17] Common Security Advisory Framework
[18] Vulnerability Exploitability Exchange
[19] Vulnerability Disclosure Report

Section 5

SBOMs in Development

5.1 Introduction - Software Development Methodologies and SBOM

Software engineering has undergone many changes in methodologies over the years. There are several different SDLCs[1] used in automotive, but some of the most common are ASPICE using the V-Model from ISO/SAE 21434 [2], agile, and waterfall. While not a full requirement, the ASPICE V-Model process implies the use of a waterfall-like methodology. Older methodologies, including waterfall, where the development lifecycle proceeds sequentially from design, through development and test, to release and operation, are still sometimes used in software engineering for industrial and operational systems.

By contrast, agile development combines requirements gathering, design, coding, test and documentation into a series of smaller iterations that eventually result in a finished software product. A major advantage of agile development is that the design is change-tolerant during the life of the project. Several related methodologies have also enhanced and complemented its effectiveness, including adoption of CI/CD[2], a practice within DevOps[3] or DevSecOps[4]. These infrastructures, architectures and techniques facilitate the security, automation, quality and repeatability of the methodology: virtual machines, containers, cloud deployments, and composable architectures. Hybrid adaptations of agile methodologies can also be compliant with ISO/SAE 21434 [2], but the development process should carefully meet agile and ISO[5] requirements.

SBOMs[6] are entirely a product of decisions made by designers and developers of software. When used in development, they provide transparency to software product composition as they evolve. For a given build/tag of software, an accompanying SBOM should faithfully represent the composition of that software. SBOMs in development offer both a primary means to assert software product compositions, and a secondary means to verify and validate conformity and quality attributes at an organization and among supply chain participants. The subtopics of this section include and expand on many of the concepts originally published in the Implementation/Development section of the 2022 SBOM-WG[7] Report. Both ASPICE using the V-Model and agile methodologies are fully compatible with SBOMs.

5.2 Actionable versus Complete SBOMs

A minimal set of information is required for an SBOM to be actionable.

5.2.1 Actionable SBOMs

The SBOM-WG member discussions and exercises suggested that to be most useful, an SBOM should be actionable. This emerging term means that its components can be uniquely identified as a particular module of binary or source code. For this document, an actionable SBOM is one that reflects the underlying software component in a standard way, using the package URL (PURL) or the equivalent. Thus, actionable SBOMs have the data needed to perform important analyses, including SCA[8] and vulnerability analysis. By analogy to medicine, the actionable SBOM is the patient chart / medical history. The patient chart / medical history has the key information that medical staff need to inform their work with patients.

[1] Software Development Life Cycles
[2] Continuous Integration and Continuous Delivery
[3] Development and Operations
[4] Development, Security, and Operations
[5] International Organization for Standardization
[6] Software Bills of Materials
[7] Auto-ISAC SBOM Working Group
[8] Software Composition Analysis

5.2.2 Complete SBOMs

Complete SBOMs, meaning the listing of every component of a potentially huge included module, are often large, awkward, and ill-suited for supply chain security efforts. It is possible for complete SBOMs to not be actionable. Component identifications at a low-level, for example individual file contents of a well-known and globally identifiable software component, but lacking the component coordinates themselves, can obscure key identifications and hinder vulnerability analysis and SCA efforts. In many cases in software, low-level identifications do not have a mapping to higher-level components that are well-known. In any case, these low-level identifications of components applied over a system comprise a complete SBOM. By documenting cryptographic hashes for every source item, they are useful for auditing and engineering purposes for recording the exact content and origin of the software delivery. By analogy to medicine, the complete SBOM is the full DNA sequence of the patient. The complete genome has obvious utility in specialized circumstances, but there is no need to bring it along to most medical appointments.

5.2.3 Usage Recommendation

Based on the SBOM-WG discussions and exercises, it is ideal if both types of SBOMs are generated and archived at the appropriate intervals. Further, actionable SBOMs using traceable IDs (such as the PURL) will be best for effective SBOM-based collaboration between supply chain members. By their nature, complete SBOMs are less useful for SBOM-based collaboration between supply chain members. Actionable SBOMs should be generated and archived for every individual build / tag. Complete SBOMs are useful for engineering and contractual / regulatory compliance reasons, and should be generated and archived, particularly for the GA[9] releases.

5.3 Automotive Industry Considerations

As outlined in §3 Ways the Auto Industry is Unique, it may be difficult for automotive industry suppliers and manufacturers to immediately transition to the recommended development practices described in this section. In many cases, a phased approach will be necessary. Moreover, the heterogeneous nature of automotive hardware and software, and the diversity of supply chains imply that multiple technologies will be used. Thus, development considerations, support, contractual obligations and other factors will influence technology selections. The criteria for selection of an RTOS[10] for a safety-critical Electronic Control Unit (ECU) may differ from that of a general-purpose OS[11] for a higher level, non-safety-critical ECU, such as an IVI[12] system. Correspondingly, target device constraints on the safety critical ECU may suggest low-overhead compiled languages for custom development. However, on a more powerful ECU, developer skillsets and productivity goals may dictate higher-level languages, libraries, and runtimes, if they conform to engineering constraints on the target. This Report anticipates the full range of systems, from tiny embedded systems to larger, higher-level more general-purpose ECUs, and the varying software selections they indicate.

5.4 Software Provenance

It is important that the SBOMs accurately reflect the software origin for the components. Software dependency provenance, agreements, licensing, and regulation should inform automated software product compliance evaluation. To effectively use SBOMs in development, the SBOM-WG suggests the following practices will ease adoption of SBOMs for development teams. The practices are listed

[9] General Availability
[10] Real-time Operating System
[11] general-purpose Operating System
[12] In-Vehicle Infotainment

in order, from highest accuracy and automation to lowest. (Note: Adoption of these practices is aided by tools.)

5.4.1 Practice 1: Standardized Software Provenance Using Automation and PURL

Many modern software development languages and tools provide native facilities for inclusion and reporting of third party components in the code base, sometimes known as declarative dependency management, see Figure 5.1 for a Java / Maven & Compatible example. SBOM development using these built-in tools will provide the most accurate, current, and complete SBOMs, by listing each component's globally unique identifiers within the programming environment. The information in the dependency listing will be accurate, by virtue of its dual use for building/running and SBOM generation. Moreover, the quality of tooling for SBOM generation from native declarative dependency declarations is high. Finally, languages with package ecosystems supporting declarative dependency management are often considered more productive, maintainable, and safe than languages lacking these facilities.

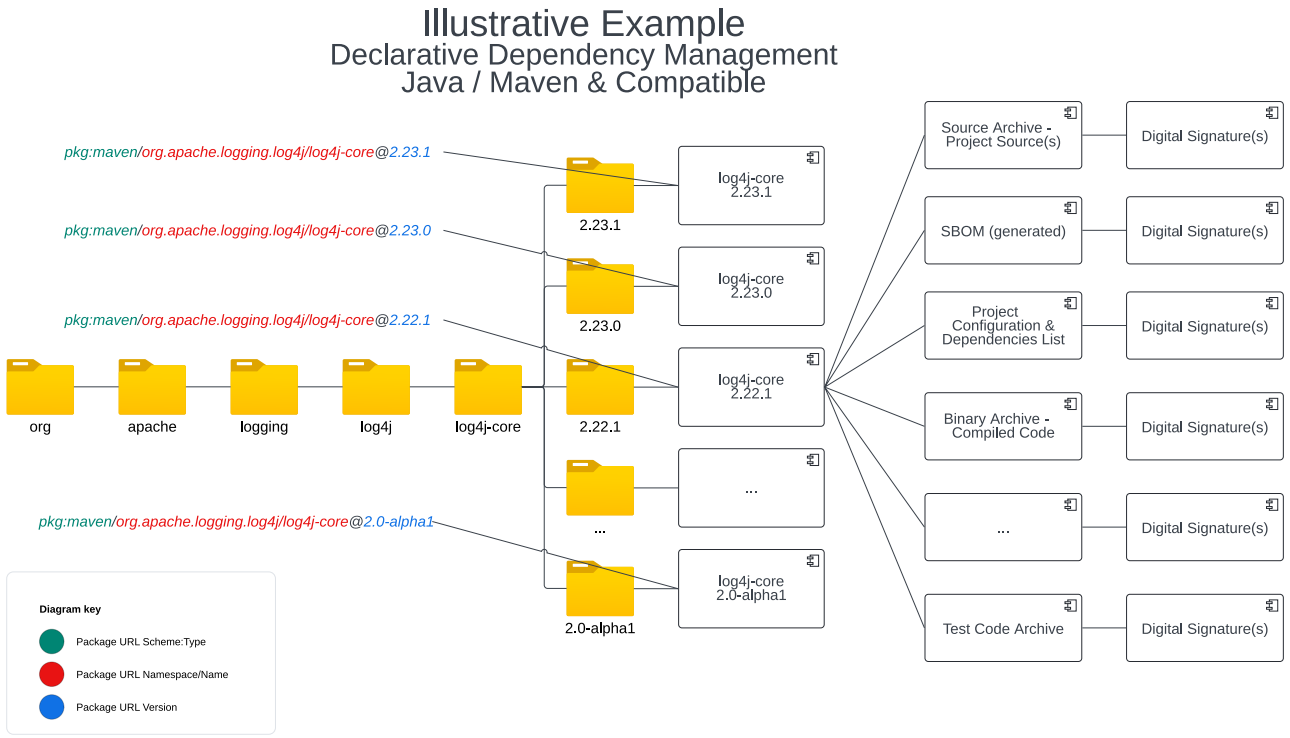

Figure 5.1 Declarative Dependency Management

SBOMs generated using declarative dependency management can be stored and managed as binaries in a binary repository using the same name and version as the code they describe. Even though SBOMs are text-based and could therefore be handled like any other source code, treating them as a build artifact instead, resulting from automated generation, is more appropriate: It is therefore recommended to make desired changes in the SBOM material in the project source code and configuration, yielding updated generated SBOM material. By treating the SBOM as a build artifact, instead of source code, the development teams ensure consistency between their project source code and configuration, and the SBOM material generated from their project configuration and dependencies. In this methodology, dependencies are identified by their coordinates, in such a way that they can be correctly retrieved through the Central Repository (or a mirror) for the ecosystem. The dependency coordinates are directly mappable to the PURL. An added bonus for this method is that many tools automatically evaluate components for vulnerabilities when they are pulled into the build. In many cases

identification of existing vulnerabilities will be a requirement from customers for suppliers. Using declarative dependency management and appropriate development tools can offer complete automation of pulls, builds, security scans, and SBOM creation.

5.4.2 Practice 2: Auxiliary Software Provenance Using Ecosystems and Tools

Non-standardized software provenance with auxiliary declarative dependency management should be used when practice (1) is not available or not practical. This method preserves some of the benefits of declarative dependency management, while forgoing more modern, memory-safe language and ecosystem options. There may be other identifier types needed, components may need manual intervention or verification, or the process may have other disadvantages compared to some of the most advanced secure build automation systems. Examples:

- **bitbake**: Embedded build and dependency manager from OpenEmbedded.

  https://www.openembedded.org/wiki/Main_Page

- **buckaroo**: C/C++ dependency manager from Meta.

  https://buckaroo.pm/

- **conan**: C/C++ dependency manager from JFrog.

  https://conan.io/

- **vcpkg**: C/C++ dependency manager from Microsoft.

  https://vcpkg.io/

5.4.3 Practice 3: Opaque Software Provenance, with Automated SBOM Generation

When automated SBOM generation is used without declarative dependency management, an SBOM can be constructed by examination and analysis of development materials after the build, also known as “forensic” analysis. By examining project source code and configuration, and sometimes by performing binary analysis, software analysis tools can construct an SBOM. SBOMs developed using this methodology should be named and versioned as binary build artifacts to match the software they describe, to ensure maximal cohesion with the software products they represent.

5.4.4 Practice 4: Manual SBOM Generation and Maintenance

When automated methods are not available for SBOM generation, this last-resort option may be employed for SBOM generation. Even lacking automation, tools that support SBOM creation can be used as assistive devices to increase accuracy in SBOM component and dependency relationships and correctness in selections and adherence to SBOM format specifications. SBOMs developed in a non-automated fashion should be versioned using source code / line-oriented / text-based version control methods.

Section 6

Exchanging and Collaborating with SBOMs

6.1 Introduction

The exchange of SBOMs[1] is a critical process in software development and cybersecurity, serving as a comprehensive record of the various components that make up software. Motivations for exchanging SBOMs include enhancing transparency, facilitating compliance with licensing requirements, and improving vulnerability management. Methodologies for exchanging SBOMs involve establishing standardized formats for documentation, ensuring secure transmission protocols, and implementing verification processes to maintain the integrity of the information exchanged. There are some concerns, however that exchanging SBOMs increases some risks that should be managed. This section aims to provide information which could be useful in designing a policy on how SBOMs may be exchanged.

6.2 Motivation

There are many reasons why an organization may wish to share SBOMs. An SBOM can be used for sharing vulnerability information with partners and customers, for license compliance, regulatory compliance, and for transparency within the organization and with its partners and customers.

6.2.1 Sharing Vulnerability Information

Prompt notification from suppliers to customers is crucial for effective vulnerability management. Information can be shared much more rapidly if automated in a standard machine readable format. Both CycloneDX, an OWASP[2]-defined format, and the CSAF[3], from the BSI[4], define a standard format by which vulnerability information can be shared. These formats are robust enough to include necessary details for the triage of a vulnerability and are extensible to support proprietary information in the properties field. Vulnerability management is discussed in detail in §7.

6.2.2 License Compliance

Using OSS[5] is a common and accepted practice for software development. However, the use of OSS comes with obligations defined by the license of the OSS component. Common obligations include attribution and reciprocity. Attribution typically involves ensuring that the copyright statement and license text be delivered to the recipient of the software. An SBOM can be a convenient container to transmit each of these. Reciprocity refers to publishing changes to the software. This obligation is associated with more restrictive licenses sometimes referred to as copyleft. While it may be impractical to deliver the actual source code, a source code URL[6] may be delivered with an SBOM. It should be noted that over time these URLs may become invalid which presents a problem outside of the scope of this document.

6.2.3 Regulatory Compliance

Note *The contents of this section rely on this analysis [3]*

A number of governments and their agencies promote the use of SBOMs either by recommendation or by regulation. CISA[7] and NHTSA[8] recommend the use of SBOMs. EO[9] 14028 “Improving the

[1] Software Bills of Materials
[2] Open Worldwide Application Security Project
[3] Common Security Advisory Framework
[4] Federal Office for Information Security - Germany
[5] Open Source Software
[6] Uniform Resource Locator
[7] Cybersecurity and Infrastructure Security Administration
[8] National Highway Traffic and Safety Administration
[9] Executive Order

Nation's Cybersecurity," directs the National Institute of Standards and Technology (NIST) to develop guidelines for the creation and adoption of SBOMs (see §2.2 United States Government).

6.2.4 Internal Transparency

It is important to share information with those outside of an organization, but also within it. SBOMs can provide a common format to describe the software being developed through the different phases of development from design to deployment. Additionally, SBOMs are living documents which can be updated as the development of a product continues to provide a single source of truth for the metadata of a component. SBOMs can also provide a means for quality assurance in several areas, most obviously in vulnerability reductions both before and after release, but also for specific testing and assurance.

6.3 Risks

The exchange of SBOMs carries inherent risks that organizations must carefully consider. SBOMs provide detailed information about the components that make up software, including open-source libraries, proprietary code, and third-party dependencies. While transparency enhances vulnerability management and compliance tracking, it also exposes sensitive information that could be exploited by malicious actors. For instance, detailed SBOMs can reveal specific versions of components that are known to be vulnerable, giving attackers a roadmap to exploit these weaknesses. Additionally, the exchange process itself can be a vector for attack if not properly secured, potentially allowing SBOMs to be intercepted or tampered with. Organizations must ensure that SBOMs are exchanged through secure channels and that access to them is tightly controlled. Furthermore, there is a risk of legal exposure if SBOMs include proprietary or licensed components without proper authorization.

6.4 Level of Detail

SBOMs can be constructed with multiple levels of detail. With less detail, an SBOM can be shared appropriately with more parties. A low detail SBOM may contain only the names of the top-level components as well as the copyright and license text. This SBOM could be used to satisfy attribution requirements common in open-source licenses. A robust SBOM which details all information about a product could include complete component details down to the function and provenance level.

6.5 Stakeholders

6.5.1 Internal

SBOMs can be useful for stakeholders within a company, particularly between the security and development teams. SBOMs can provide a standardized means by which component name and version information may be shared. In addition, SBOMs can be a useful artifact for ensuring quality, integrity, and legality of the software produced. Quality checks such as scanning for the MISRA[10] violations or CWEs[11] is an important part of software development. Placing evidence of these scans in an SBOM enables a company to easily show that due diligence has been performed in minimizing the defects present in software. While an entire report may be stored in the SBOM, consider the alternative of storing the report URL and cryptographic hash.

[10] Motor Industry Software Reliability Association
[11] Common Weakness Enumerations

6.5.2 External

While it is recommended that SBOMs be carefully protected as with other valuable IP[12], it may sometimes be necessary to share SBOM data to parties outside the supply chain. In the future there may be circumstances where SBOMs are required to be shared with third parties. In general, it is recommended that, due to increased risk of IP and cybersecurity exposures, automotive SBOMs should not be shared outside the supply chain partners. This Informational Report ("Report") suggests that SBOMs should not be shared with retail customers, the general public, or other third parties unless required.

6.6 Transmission Methods

Sharing SBOMs internally carries much less risk than sharing them externally, and many workflow, messaging, and developer tools will easily and in some cases automatically allow sharing of SBOMs.

The method by which SBOMs are shared externally depends on the sensitivity of the information within. For a minimal SBOM with no confidential information, if allowed by agreement between customer and suppliers, the SBOM may be shared freely using email, file sharing, and other tools up to and including public posting on a website. SBOMs meant only for distribution to an organization's partners and customer should be shared only through a secure channel with end-to-end encryption. Secure channels and file exchange are common in the automotive industry. Encrypted and signed files can be exchanged easily on an ad hoc basis via email, file sharing, and other collaboration tools.

One especially straightforward method is to simply use the distribution methods already in place for other sensitive and proprietary data. Many supply chain participants in the industry exchange secure documents via a web portal system with specific user authorizations. Documents, including SBOMs, that are exchanged in this way cannot only be verified as secure, authentic, and available, but also become a permanent part of the project records. The transmission method may be specified in the CIA[13] (described in §4.4 Cybersecurity Interface Agreement and Development Interface Agreement), especially if strong access control and user management is needed. Pre-agreement between supply chain participants for any SBOM terms is an important consideration for any sharing between supplier and customer.

6.7 Integrity Checks

Integrity checks are critical for ensuring that the SBOM has not been altered by malicious actors or transmission errors. The two most common ways of ensuring integrity in SBOMs are hashes and digital signatures. Hash algorithms such as SHA[14] do not provide non-repudiation, but are robust against errors in transmission. Digital signatures can prove authorship, but the keys used to sign the SBOMs must be managed carefully through PKI[15] or some other secure method. It is up to the organization to weigh the costs and benefits associated with guaranteeing authorship.

6.8 Sharing SBOMs During Product Development

Any software project and support methodology will support SBOM sharing. The following uses an example waterfall model similar to the ISO[16] V-model, as described in the Auto-ISAC[17] Security De-

[12] Intellectual Property
[13] Cybersecurity Interface Agreement
[14] Secure Hash Algorithm
[15] Public Key Infrastructure
[16] International Organization for Standardization
[17] Automotive Information Sharing and Analysis Center

velopment Lifecycle (SDL) BPG[18]. As defined by the Auto-ISAC SDL BPG, there are six key phases in the development of a new automotive product:

1. Pre-development (architecture design)
2. Requirements
3. Design
4. Implementation
5. Testing and verification
6. Post-development and production operations

In this section, we will introduce each of these phases, show how SBOMs pertain to them, and illustrate possible outcomes and considerations. The processes described below show an example for building and sharing SBOMs within a development project. Processes at a given supplier or customer may differ. For example, an agile software development process may require sharing SBOMs at different times than waterfall development. Whichever SBOM process is ultimately adopted, the Report recommends that practices be consistent across development teams as a policy for the company.

6.8.1 Pre-development (Architecture Design)

The first phase of the SDL considerations focuses on determining the level of risk that is acceptable/ unacceptable for a final product. This phase of the SDL is highly independent and in most cases is an analysis by a single entity. In this phase of the SDL SBOMs cannot be generated since the hardware and software to be used are yet being chosen. Instead, the focus of this phase will be to define criteria that could be used to judge the software components to be incorporated, which is especially important for open source software. The criteria will allow entities to objectively accept/reject software and may include topics such as completeness of functionality, age of software, support team depth, maintenance status, known security issues, licensing terms, and other salient criteria.

Outcomes:

1. A defined set of rules to evaluate candidate software components to be used

Considerations:

1. Pre-existing agreements between supplier and customer may govern this process

6.8.2 Requirements

The second phase of the SDL focuses on the development of a comprehensive list of cybersecurity requirements for a final product. Unlike the first phase of the SDL, this phase is highly collaborative (between supplier and customer) and will require significant input from all the parties involved. In this phase of the SDL, the content and detail of the SBOM to be shared should be defined. Also, in this phase, the parties will develop an agreement on the previously defined criteria for software evaluation.

Outcomes:

1. Any rules, requirements, or guidelines for SBOMs agreed upon by the supplier and customer
2. Detailed specifications governing SBOMs and software components

[18] Best Practice Guide

Considerations:

1. The first version of an SBOM may be generated during this phase, particularly if a hardware-specific software library or firmware is specified.
2. Due to the high degree of variability, it is not recommended that the SBOM be shared during this phase.

6.8.3 Design

The third phase of the example SDL focuses on the design of the supplied software. This phase may be conducted in collaboration (between supplier and customer) or individually (within supplier), and often will determine the software components to be used. In the Design phase of the SDL, enough information about the software should be available so an early version of the SBOM can be generated. As product modifications are made the SBOM should be continuously updated to reflect the latest software revisions.

Outcomes:

1. Optional initial SBOM
2. Optional sharing of initial SBOM
3. SBOM exchange methods between companies agreed, tested, and verified (may be based on existing document exchange methods or other processes as defined by the participants)
4. SBOM sharing frequency
5. Triggers agreed (e.g., at development milestones, every month/week, etc.)

Considerations:

1. While sharing SBOMs at this early stage of the SDL can be useful, suppliers and customers should be aware that there may be frequent changes and should manage any sharing accordingly.
2. Suppliers and customers should agree specifically on any sharing requirements or other SBOM specifications prior to completing the Design phase.
3. An optional consideration is to share SBOM parts that are known to be stable such as hardware component SBOMs.

6.8.4 Implementation/Development

The fourth phase of the SDL focuses on the implementation (i.e. development - note that “implementation” is synonymous with “software development” for the automotive lifecycle). It is mostly conducted by individual companies with a certain degree of collaboration between the parties at key points of the development process. In the Implementation phase of the SDL, the as-built SBOM will begin to be populated with the software being integrated. Component choices should be complete or nearly so by the time the phase ends. The Implementation phase is perhaps the most critical one in the SDL for SBOM construction, and if properly done will result in an accurate SBOM.

Outcomes:

1. Complete or nearly complete component choices by developers
2. Complete or nearly complete SBOM versions
3. Pre-release SBOM may be shared with external entities

Considerations:

1. SBOMs created during development may undergo frequent changes. Receivers of any SBOM in the implementation phase should be aware that apparent risks may be mitigated before the software is released.
2. Changes to the SBOM in late-stage or post-development may invalidate any prior analysis by either supplier or customer.
3. Applicable confidentiality procedures for SBOM data should be observed for pre-release SBOMs.

6.8.5 Testing and Verification

The fifth phase of the SDL focuses on the testing, quality assurance, verification, and integration of the supplied software. The Testing phase is typically conducted both individually (within supplier) and in collaboration (between supplier and customer) for a given product. In this phase of the SDL, the SBOM may be modified due to code changes required during the testing and verification process. At the end of this phase, SBOM content should be final or near-final. Once the coding is fully complete, SBOM updates are only needed for post-release updates and patches.

Outcomes:

1. SBOM is finalized at the same time as the final build for the software that it describes
2. SBOM should be shared per agreement between supplier and customer

Considerations:

1. Changes made during this phase, while likely minimal, should be included in the final SBOM.
2. SBOMs can be shared as part of the software fulfillment process (next phase).
3. Any subsequent release/update of a software product implies an updated SBOM as well.
4. Updated SBOMs should be shared during any subsequent release/update process.

6.8.6 Post-Development: Release, Fulfillment, and Production Operations

The sixth and last phase of the SDL focuses on the process of SBOM support for software updates after a product has been released in the field. Software can be updated by suppliers for many reasons, including bugfixes, new features, and vulnerability mitigation. Regardless of the reason for the software update, the supplier should always create a new SBOM that describes the new version of software. As part of the software update, the supplier should share that SBOM with appropriate parties.

Outcomes:

1. Supplied software modifications, updates, and releases
2. A new SBOM created for any new or modified version of supplied software and shared subject to agreement
3. Software versions and their corresponding SBOM should be tracked by each supplier
4. Suppliers and customers should continuously review components within the supplied software for vulnerabilities
5. Suppliers should provide updated supplied software and new SBOMs in response to new existing vulnerabilities
6. In addition to software updates and new SBOMs, suppliers may provide technical bulletins, controls, mitigations, workarounds, and other materials and support to customers

Considerations:

1. Length of support will likely be many years; this practice suggests the provision of long-term archives and updates of both supplied software and SBOMs.
2. Monitoring for discovered vulnerabilities is out of scope for this Guide, but it is thought that most suppliers can use existing processes with adaptations.
3. Online and background update methods, such as OTA[19] updates, will be a critical part of vulnerability mitigations.
4. Ordinarily a specific version of supplied software will have one and only one SBOM. New software versions and updates imply a new SBOM.
5. SBOM changes for a specific version of supplied software after release should be rare and limited to error corrections or to fill in missing information.
6. Multiple SBOMs for the same version of supplied software can be distinguished by the timestamp field. The correct SBOM is the one with the latest timestamp.

[19] Over-the-Air

Section 7

Vulnerability Operations Using SBOMs

7.1 Introduction

Note *The contents of this section rely on the concepts defined in ISO/SAE 21434 [2].*

Perhaps the most important reason for SBOMs[1] is to understand quickly if an item, component, or system may have a known vulnerability. With the proper tools, SBOMs enable automated, near-instant recognition of vulnerabilities in open source and third-party software, and without explicit input from the software supplier. It should be noted that vulnerabilities are not necessarily exploitable, but SBOMs can assist in automation of exploitability analysis as well as continuous monitoring.

7.2 Supply Chain Vulnerability Management

The integration of SBOMs into supply chain management is a transformative approach for implementing the guidelines in ISO/SAE 21434 [2] and UNECE WP.29 / R155 [4]. By requiring SBOMs from upstream suppliers, organizations can significantly enhance their vulnerability operations processes and their CSMS[2]. SBOM scanning enables systems integrators and their suppliers, to meet the ISO/SAE 21434 [2] requirements below:

- RC-07-02 - Supplier should support the evaluation of cybersecurity capability
- RQ-07-06 - The identification of vulnerabilities in supplier components

For suppliers, the ongoing awareness and triage of vulnerabilities for the underlying software components in their products are necessary for vulnerability management. For both customers and suppliers, vulnerabilities that are found and/or are present in software products and components need to be documented, tracked, and, if necessary, remediated. And in both cases, the action must take place with or without SBOMs, but the process can be much more efficient with complete and actionable SBOMs (see §5.2) provided in the supply chain.

7.3 Vulnerability Management - Basic Steps for Using SBOMs

- RQ-08-05 Weaknesses shall be analyzed to identify vulnerabilities

Tracking and remediating vulnerabilities is a well-known process with many different approaches, but all have a few key steps in common, summarized below:

- Communication with stakeholders
- Hardware/software inventory management
- Threat and exploitation intelligence gathering
- Review of inventory for exposure to vulnerability

7.3.1 Communication

Communication between suppliers, customers, and other stakeholders is central to any vulnerability management program. Communication activity includes internal coordination as well as appropriate disclosures to customers, researchers, the Auto-ISAC[3], regulators and sometimes to the public. Communication is not only required during an incident response and resolution. Rather, it should start from the time a vulnerability is discovered through the End of Life (EOL) of the product.

[1] Software Bills of Materials
[2] Cybersecurity Management System
[3] Automotive Information Sharing and Analysis Center

For vulnerabilities in underlying components, SBOMs facilitate timely and accurate communication by narrowing the scope to a specific module and function, and also by lessening lead time for resolution. With SBOM scanning, suppliers can provide instant notification of newly-found open source vulnerabilities to the right internal engineering teams and therefore reduce triage time. At any point a supplier can provide better information about the vulnerability to its customers by identifying a specific vulnerable component. Likewise, customers can communicate internally and back to suppliers about the specific issue and any recommended defensive measures. The SBOM-WG[4] recommends frequent and candid vulnerability communication between suppliers and customers, which requires full and complete exchange of SBOMs as a prerequisite.

7.3.2 Software Inventory Management

An accurate and detailed inventory is the foundation for many cybersecurity functions, and vulnerability management is a prominent example. Searching code for vulnerabilities during development and in delivered, post-development products poses a major challenge as searching code is very resource and time intensive. Complete SBOMs enhance software inventory management by providing a detailed list of all components in a product.

The SBOM is separate from the code and lends itself to database or tool storage, which can be purpose-built SBOM-specific application database(s) as offered by several vendors and OSS[5].

Vulnerability managers and product security teams have the responsibility in the management of SBOMs. This responsibility includes both the receipt and processing of SBOM data from suppliers as well as processing updates to SBOMs for product updates. For SBOMs from suppliers, they ensure that inbound SBOMs from suppliers are properly imported, stored, and secured into the appropriate databases. For any updates from suppliers or updates made to the product, updates to the SBOM have to be promptly executed.

7.3.3 Threat Intelligence

Traditionally, the monitoring of public and private alerts for emerging vulnerabilities, threats, exploitation, and remediation is essential for all cybersecurity operations, for both suppliers and customers. Vulnerability managers and product security teams are required to gather threat intelligence. Threat intelligence and software inventory constitute the key data for determining the presence or absence of a vulnerability in a software product. ISO/SAE 21434 [2] has the requirements supporting the use of SBOMs in the triage process:

- WP-08-01 - Cybersecurity Sources identify cybersecurity issues (they can be persisted in SBOM metadata; e.g., name-value pairs in CycloneDX metadata)
- RQ-08-03 - Cybersecurity information shall be collected and triaged to determine if a vulnerability becomes a cybersecurity event

Triage of threat data can assist in filtering a flood of event data for relevance. The amount of cybersecurity information is vast, so automation is important to effectively triage it. For example, keywords can be derived from current threat intelligence reports and include terms related to the TTP[6] used by cyber adversaries. By utilizing a set of predefined keywords, cybersecurity professionals can quickly filter through emails, private and public platform alerts, and other reports to pinpoint significant events that warrant attention. Keywords can vary by organization, and can include things like ‘lwip,’ ‘usb,’ ‘linux,’ or ‘openssl.’ Other criteria may be used for triage as well, such as the CVSS[7] score and attack vector.

[4] Auto-ISAC SBOM Working Group
[5] Open Source Software
[6] Tactics, Techniques, and Procedures
[7] Common Vulnerability Scoring System

The SBOM-WG recommends considering commercial and open source software tools to help monitor and filter threat intelligence data. Regardless of how information is triaged, automation is recommended when possible. Effectively using SBOMs is a big step towards automation.

7.3.4 Inventory Review

The next step in identifying a possible vulnerability is to match asset items with vulnerability alerts. Simply, each emerging vulnerability disclosure should be checked against current software inventory to check for a match, and any match that is found will need further processing.

Without SBOMs, monitoring of alerts by consumers is possible for products and finished software but not for individual granular components. Further, product alerts will generally be provided by the supplier only, and sometimes only after a lengthy triage process during which the customer may be vulnerable to attacks. However, vulnerabilities are often found in open source and other third-party modules. The supplier of the product must then perform triage to understand if the product is vulnerable to the flaw in the component. Once the determination is made by the supplier, customers can be notified.

With SBOMs, combined with proper monitoring of threat intelligence, the customer and supplier can become aware of vulnerabilities at the same time, and both can accelerate their defensive tactics.

7.4 SBOMs and ISO/SAE 21434 Section 8 - Continual Security Activities

Whether structured or unstructured, SBOMs are an integral part of cybersecurity monitoring. ISO/SAE 21434 [2] Section 8 closely mirrors the vulnerability management process described above, although that specification does not specifically mention SBOMs.

- RQ-08-07 - Vulnerabilities shall be managed to assess risks and their treatment or remediated

As a result, SBOMs are especially suited to support of Section 8 and the ongoing cybersecurity activities it prescribes. In addition to keeping a list of components, SBOMs can also store several ISO/SAE 21434 [2] Work Products [2], such as cybersecurity sources, triggers, cybersecurity events, and vulnerability management. The SBOM-WG recommended SBOM formats, the SPDX[8] and CycloneDX, support inclusion of Work Product [2] data.

7.4.1 SBOM Scanning for Vulnerabilities

The integration of SBOMs into continuous monitoring operations is also a transformative approach for implementing the guidelines in ISO/SAE 21434 [2] and UNECE WP.29 / R155 [4]. By incorporating SBOMs, organizations can signicantly enhance their vulnerability operations processes and their CSMS. Continuous monitoring, supported by SBOM scanning, allows for rapid response to new threats, aligning with the dynamic nature of cybersecurity and the requirements of ISO/SAE 21434 [2]. ISO/SAE 21434 [2] does not specifically prescribe SBOMs, but it has recommendations and requirements that are well-supported by using SBOMs.

7.4.2 Low Risk Vulnerabilities

Vulnerabilities that are deemed low-risk or not exploitable still must be tracked as part of the continuous monitoring required by ISO/SAE 21434 [2]. The cadence for checking for low-risk vulnerabilities that have become exploitable or higher risk can be at regular intervals, or at project milestones. A better solution is to have an SBOM scanning tool able to continuously scan or monitor existing low-risk vulnerabilities that become exploitable or higher-risk (see §7.4.1).

[8] Software Package Data Exchange

If a vulnerability is determined to be exploitable or otherwise becomes a high-risk item, the vulnerability needs to be managed, which includes being tracked and mitigated. Exploited vulnerabilities become cybersecurity incidents or events.

7.5 Incident Response - SBOM Use

A cybersecurity incident or event occurs when a vulnerability is exploited. The purpose of the Auto-ISAC is to provide a safe environment in which to share information on exploits that can be very complex. When a vulnerability is found, information sharing among affected parties is a useful and effective practice. The use of SBOMs is an extension of information sharing that provides insight into the vulnerabilities that are present. ISO/SAE 21434 [2] requires the handling of cybersecurity incidents with WP-08-03 Cybersecurity Events.

Common steps in the incident response process are:

- Establishing triggers for possible incidents
- Triage incidents and perform root cause analysis
- Ticketing and dispatch of vulnerability remediation decision and activities
- Remediation action for vulnerable items

7.5.1 Triggers

As covered above, ISO/SAE 21434 [2] Section 8 requires continuous monitoring. Specifically, Work Product WP-08-02 requires the establishment of triggers. A trigger is an event or situation that indicates a vulnerability may be subject to an exploit, otherwise known as an incident. Automation of incident notifications using SBOMs is possible using emerging alerting tools such as the CSAF[9], which are machine-readable and can act as triggers for automated vulnerability resolution processes. An even more useful capability of automated notications is to provide the VEX[10] assertions. These assertions enable suppliers to advise customers on known vulnerability status and exploitability. Given the historically high fraction of vulnerabilities that cannot be exploited, VEX is a powerful technique for saving both supplier and customer from time-consuming and unnecessary remediations.

7.5.2 Triage and Evaluation

Because the large number of possible threats and vulnerabilities can make follow-up difficult, it is reasonable for an organization to prioritize events by following a structured risk assessment process. Once a relevant vulnerability is identified, the organization must assess the likelihood of a cybersecurity event occurring and the potential impact on its products and customers. This involves analyzing the severity of the vulnerability, the complexity of exploitation, and the potential damage to confidentiality, integrity, and availability of data. The organization should then prioritize the risks based on their score, which is a combination of the likelihood and impact assessments. Remediation strategies must be developed for incidents, which may include patch management, code updates, and customer notifications. Continuous monitoring for new vulnerabilities and threats is essential, as is a robust incident response plan that outlines steps for containment, eradication, and recovery in the event of a security breach. This proactive approach ensures that the organization can quickly respond to and mitigate the effects of cybersecurity events on its open-source components. Automation of triage and risk assessment can be extremely helpful for both suppliers and customers. Vulnerabilities identified during SBOM scanning help to rapidly identify affected systems and evaluate exploit severity.

[9] Common Security Advisory Framework
[10] Vulnerability Exploitability Exchange

7.5.3 Ticketing and Dispatch

Any vulnerabilities that are found in development or product software should ideally be logged via ticketing and dispatched for follow-up and resolution. However, in practice there are often far too many possible vulnerabilities, so some organizations prioritize those that have the most potential for harm.

Regardless of the qualifying criteria, for any vulnerability that requires further action, it is recommended to use a ticketing or tracking system to ensure there is a record of the issue and any actions taken for remediation. Triage of the vulnerability is again required to properly dispatch the ticket to the appropriate team for the actions, and the remediation team should note progress and resolution of the issue in the ticket.

SBOMs are helpful in the ticketing and dispatch workflow by providing information on the software content of the product and referencing the CVEs[11] for details. SBOMs narrow the scope of remediation activities by focusing on fixes for the underlying component rather than the higher-level software product.

7.5.4 Remediation

In most cases, remediation for an affected product or component requires the responsible engineering team to make a change. In many cases, the SBOM can easily show the component that needs updating or patching, and the engineer can simply replace the vulnerable component with a patched or updated one.

Developers should also update any associated SBOM and distribute it to customers and internal organizations according to agreed and established processes. It is also important to ensure there is transparent identification of any affected software by using a clear naming and versioning system. In particular, semantic versioning is a simple and effective model that can support SBOM-based vulnerability management.

Once the development team has made the necessary changes to the code and to the SBOM, the change should be propagated to the affected systems. That process is essential but because it is not strictly related to SBOMs it is out of scope for this discussion. The only aspect of the process that is worth noting is again that updated SBOMs that match the new software should be available at the same time as the software itself.

7.6 Cautions for SBOM-based Vulnerability Management

While SBOMs can make vulnerability management significantly more efficient, they are not a panacea. There are several important cautions when using SBOMs. Problematic identifications cause faulty analyses; these identifications may improve over time. Missing software development standardization, nuanced individual software packages, and other maturity gaps contribute to analysis deficiencies. The SBOM-WG recommends that automotive product security teams diversify their sources and tools to address the limitations of vulnerability identification and mapping.

7.6.1 False Positives

Depending upon the developers' implementation of a given component, the SBOM may not provide enough information to distinguish between an actual vulnerability and the apparent presence of one. For this reason, SBOMs cannot be used to automatically detect vulnerabilities, but instead automatically trigger processes for further investigation.

[11] Common Vulnerabilities and Exposures

That can happen when a developer does not use the whole library but uses only a subset, or when a vulnerable section of code is not executed by the software in operation. A vulnerability alert can be issued for a given component that has nothing to do with the subset of the code that is active in a supplier's product. VEX (see §7.5.1) can help manage issues like this.

7.6.2 False Negatives

Tools for mapping SBOMs to vulnerabilities require some knowledge base and real-time feeds for identifying vulnerabilities. Limitations on the data in these knowledge bases and feeds can result in a failure to recognize real vulnerabilities. Diversifying vulnerability sources will help ensure accuracy and accelerate vulnerability detections for vulnerabilities that are not widely published.

False negatives are almost guaranteed for unreleased but known supplier vulnerabilities and, worse, zero-days. Responsible suppliers will generally try to issue patches ahead of general vulnerability alerts so their customer will have an opportunity to apply patches before exploits can be launched, so a vulnerability can exist for some time without disclosure. In the case of zero-days, constant diligence and quick reaction is the only solution.

7.6.3 Inaccurate Identification

SBOMs can be created a number of ways (see §5) and the only one that approaches guaranteed accuracy is an automated build program that includes declarative dependency management. Studies on static analysis tools have not only shown major inaccuracies for identification of included components, but also found wide differences between the inaccuracies themselves, and even found phantom components that were not actually included at all.

7.6.4 Non-standard Identification

Components can be identified in many ways depending upon, variously, the author or supplier, the open source ecosystem, the language ecosystem, the repository, and many others. There is no required standard for a particular identifier for any software and in fact identity has been cited as one of the hardest problems in computing.

For the automotive industry, a major gap in SBOM tooling is the large number of components purchased from commercial software component suppliers. Scans do not easily reveal some of these common utilities and component databases including automated build systems treat them differently from code that is sourced from well-known repositories.

7.7 Conclusion

SBOMs are a powerful tool for timely awareness and diagnosis in vulnerability management. Extensions of SBOMs in the future, including machine-readable alerts and automation as well as more advanced automated communications such as VEX, have great promise for faster and less labor-intensive vulnerability management.

SBOM-based vulnerability management is still in early stages. However, increased supply chain attacks and consequent trends in industries, standards bodies, and governments suggest that it is an important and necessary improvement in what is today a difficult manual process. And while tooling is also new, vigorous vendor competition has brought a wide array of constantly improving tools. For all those reasons, the SBOM-WG recommends that SBOM-based vulnerability management programs should be actively evaluated and implemented in the automotive industry.

Section 8

Review of SBOM Workshop Exercises and Findings

8.1 Introduction

Two face-to-face workshops were organized in 2023 with the specific goal of conducting a detailed and practical examination of key use cases for SBOMs[1] within the context of the Auto-ISAC[2] and member organization's operations. During the first of these sessions, multidisciplinary teams engaged in a series of tabletop exercises to game out and delineate several crucial SBOM use cases. Participants were expected to navigate general workflows of identified use cases with an understanding of their organization's methodologies. The teams articulated potential challenges that could emerge throughout the SBOM lifecycle. It was recognized that tooling would play a vital role in the efficient execution of SBOM use cases. The second workshop provided a platform for SBOM tooling vendors to showcase their solutions with a 10-minute overview and complementary exhibit and Q/A session. Despite their presence, there was a strict policy to neither use nor endorse any specific tools or vendor during the event to maintain impartiality and focus on the exercise's educational objectives. The second workshop also included a breakout session with additional SBOM tabletop exercises.

8.2 Key Insights

- Although the use of real SBOM samples was considered for both workshops, the emphasis remained on the analytical and strategic value derived from the team exercises.
- Critical examination of how various stakeholders in the development chain, including the Original Equipment Manufacturers (OEMs), the Tier-1s, the Tier-2s, and government entities, would utilize an SBOM, was a focal point of the discussions.
- The entire lifecycle of the product from early development, production, ongoing scanning and patching through end-of-life were a key focus for use cases - with the required flow of data between the stakeholders considered.
- Automated and manual tagging is required for resolution, non-applicability, other compensating controls, or risk acceptance within the SBOM. This valuable metadata must carry throughout the lifecycle.
- Automation is absolutely necessary, but tooling may not support a robust SBOM lifecycle at present.

8.3 SBOM Workshop - Round 1

The first face-to-face SBOM workshop took place on 2023-01-26 in Farmington Hills, MI. The meeting was hybrid, including both in-person participants and online members. The participants engaged in 3 tabletop exercises focused on SBOM use cases. The exercises were as follows:

- Exercise 1: Creating an SBOM
- Exercise 2: Transferring SBOMs between parties
- Exercise 3: Managing SBOMs over time

8.3.1 Key Use Cases Explored

- Integration and Delivery: Participants explored assembling a sample embedded product, with dependencies built by the Tier-2 suppliers, incorporated the Tier-1 suppliers, and ultimately delivered to the OEM.
- Transparency and Collaboration: Participants explored information sharing through SBOMs, discussing the visibility level needs of the OEMs, the Tier-1, and the Tier-2 suppliers.

[1] Software Bills of Materials
[2] Automotive Information Sharing and Analysis Center

- Upgrade Cycle Testing: Participants simulated an upgrade cycle to evaluate the efficacy of SBOMs in managing and tracking changes and dependencies during product updates.

8.3.2 Key Findings

As a result of the workshop, several new SBOM topics and pain points were identified. These include:

- Identifying SBOM creation metadata (such as time of creation vs CVEs[3])
- Inclusion of non-FOSS[4] components, especially non-public proprietary Tier-1 components, needs a means to cleanly integrate - even if most of the OEMs won't track these as closely (since they may not have public CVEs)
- It is very common in embedded systems to have components that aren't used or aren't used in the same manner CVE ratings expect - meaning a means to manually change the risk or provide commentary on why the issue is not applicable is needed. This should ideally be integrated within the tool.
- License compliance vs CVE detection
- Security vulnerability analysis related to SBOM is an activity unique to itself.
- Handling and managing extra information and references within SBOMs
- Report generation for CVEs, CWEs[5], etc.
- SBOM quality assurance metrics
- Automated and manual tagging is required for resolution, non-applicability, other compensating controls, or risk acceptance within the SBOM. This valuable metadata must be carried throughout the lifecycle.
  - This is especially important if SBOMs are required to be shared in some form through the value-chain (including regulators) as the true story of the risk cannot be determined otherwise.
- Automation is absolutely necessary, but tooling may not support a robust SBOM lifecycle at present:
  - Current tooling does an adequate job in discovering public FOSS libraries and tying them back to the license.
  - Licenses are static for a given component/version; CVEs are not.
  - Tooling mapping the discovered component to the CVEs is marginally correct at a point-in-time only.
  - Managing the linkage and tracking patching through the product lifecycle is not well handled.
- Linking SBOMs with device roots of trust may become possible and desirable in the future. Today, hashes are used in the standard formats to link an SBOM back to the software it represents. In a possible future zero trust model, software, roots of trust, and transparency data like SBOMs may be able to be cryptographically linked together.
- SBOM lifecycle and versioning - this becomes especially challenging as versions and patches of the software accumulate, especially as the software forks through the life of the product.

Overall, the workshop was a productive experience and illustrated the complexity of using SBOMs in the automotive industry. These exercises provided critical insights into the operational application of

[3] Common Vulnerabilities and Exposures
[4] Free and Open Source Software
[5] Common Weakness Enumerations

SBOMs and highlighted the areas where improvements are necessary for the SBOM framework to be effectively integrated into the product lifecycle. The understanding gained from these activities will inform the development of strategies for SBOM deployment, tool selection, and process optimization.

8.4 SBOM Workshop - Round 2

The second face-to-face SBOM workshop took place over two days on 2023-05-17 and 2023-05-18 in Farmington Hills, MI. The meeting was hybrid, including both in-person participants and online members. On the first day, participants attended tool vendor introduction presentations and participated in question-and-answer sessions. The workshop also offered a small expo area for vendors to demonstrate their products. The second day focused on SBOM tabletop exercises and discussions. Two tabletop exercises were conducted. The morning session focused on creating and naming SBOMs, while the afternoon session focused on validating and analyzing SBOMs.

8.4.1 Key Use Cases Explored

- SBOM Creation and Progression: Participants discussed the process to create an SBOM from source or binaries and determining subsequent actions required to advance the lifecycle of the SBOM.
- Naming Conventions and Challenges: Participants discussed the complexities of naming an SBOM or its components and addressing the challenges associated with consistent identification within the automotive ecosystem.
- SBOM Validation: Participants identified the need to develop methodologies to ensure that SBOMs accurately reflect the component list and align with an arbitrary checklist of components, versions, and vulnerabilities.
- SBOM Analysis: Participants discussed comparing and decomposing multiple SBOMs, including those of related products or different versions, to understand the relationships and challenges of a deep supply chain.

8.4.2 Key Findings

As a result of the workshop, several new SBOM topics and pain points were identified. These include:

- There is a requirement for precise terminology among the Auto-ISAC members, stakeholders, and tool vendors
- A defined vulnerability format for private vulnerabilities within an organization and in the Auto-ISAC (similar to a CVE) could be defined for tool compatibility

Section 9

Tools

9.1 Introduction

SBOMs[1] applied to automotive software are a complex topic. The tools highlighted in this section range from simple, single-purpose applications or utilities to the much more complicated infrastructures that are needed. These infrastructures support the complexities of secure distributed development across the automotive supply chain. Many of the tool types profiled are offered as products, while some are additionally or alternatively offered as services. No matter how these functions are realized, it should be expected that operationalizing the kind of enterprise / cloud software between organizations and providers to facilitate this development is an involved undertaking.

The SBOM-WG[2] recommends looking beyond older techniques used for file-sharing, software design, development, testing and distribution, SBOM production and dissemination, and other functions. Instead, the SBOM-WG has discussed the application of modern and advanced methodologies and infrastructures that directly support a secure supply chain, partly by their rigorous application of SBOM technology.

9.2 Automotive Product Development Process - A Prescriptive Method

Automotive developers often write software (code and configuration) using external dependencies to satisfy use-cases / product feature requirements. Compiled languages are compiled (translated to bytecode) and sometimes linked (bytecode arranged for memory loading and execution) for a target platform. A build step generates one or more SBOMs for the software using the project configuration, and sometimes the source code. SBOM documentation of those distributable binary/source artifacts document the provenance of the software, including the dependencies. After testing, the binary artifacts (compiled languages) and source artifacts (interpreted languages) are the basis for the software portion of automotive products. The SBOM documentation of the software provenance is used to inform authorized supply chain participants of the structure and origin of the software. Moreover, the SBOM documentation serves as a basis for distributed supply chain members to apply C-SCRM[3] to the software product. The steps described are repeated, often indefinitely until the End of Life (EOL) as feature changes and additions, and other software updates and fixes are accommodated.

9.3 The ISO 26262 V-Model

The ISO 26262 standard defines the V-Model as depicted in Figure 9.1.

9.4 Effective Practices

Some general techniques that apply to the tools and technologies of the following sections, particularly single-purpose tools include:

- Use of multiple tools
- Consensus-based results from tools
- Primary methods and supporting evidence

9.5 Identity and Access Management

The IAM[4] function manages access using user-role based permissions for the individuals and systems performing work in SBOM-related activities. Due to the distributed nature of the supply-chain and the

[1] Software Bills of Materials
[2] Auto-ISAC SBOM Working Group
[3] Cybersecurity Supply Chain Risk Management
[4] Identity and Access Management

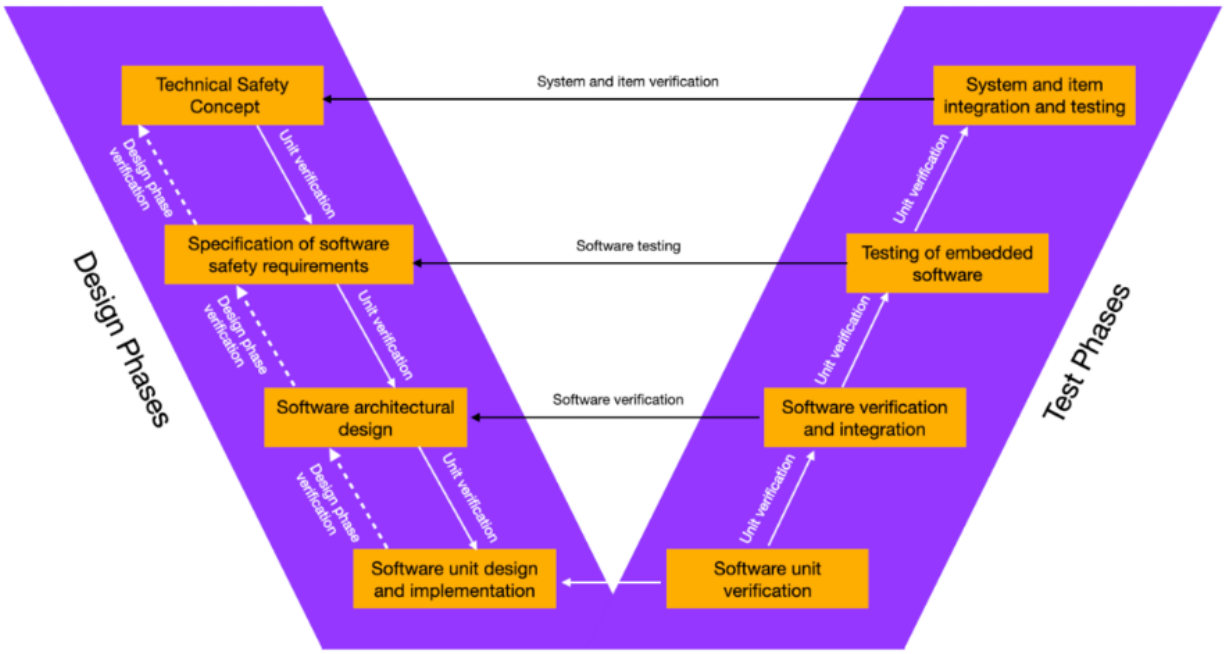

Figure 9.1 ISO 26262 V-Model (https://about.gitlab.com/solutions/iso-26262/)

task, older single-directory/single-protocol solutions are unable to fulfill all the needs envisioned. For example, in SBOM-related tasks, these may include:

- Provisioning / Deprovisioning accounts for requirements writers, developers, testers and others needing identities that span organizations
- Authenticating and authorizing a developer, tester, or product or project manager to PM/issue/feature-tracker
- Authenticating and authorizing a developer to a source repository / source control system
- Authenticating and authorizing a developer or a CI/CD[5] job to publish build artifacts like binaries and generated SBOMs to a binary repository
- Federating on-premise enterprise systems with cloud-based offerings

Relevant protocols, schemes and technologies include:

- OAuth[6] 2.0 https://www.rfc-editor.org/rfc/rfc6749
- OIDC[7] https://openid.net/specs/openid-connect-core-1_0.html
- SAML[8] 2.0 https://www.oasis-open.org/standard/saml/
- FIDO[9] Alliance https://fidoalliance.org/specifications/download/
- MFA[10] https://www.cisa.gov/resources-tools/resources/multi-factor-authentication-mfa
- SSO[11] https://www.cloudflare.com/learning/access-management/what-is-sso/
- RSO[12] https://www.gartner.com/en/documents/1405032

[5] Continuous Integration and Continuous Delivery
[6] Open Authorization
[7] OpenID Connect
[8] Security Assertion Markup Language
[9] Fast Identity Online
[10] Multi-factor Authentication
[11] Single Sign-on
[12] Reduced Sign-on

9.6 Binary Repository

The binary repository function archives binary artifacts, as opposed to line-oriented source artifacts. More sophisticated binary repositories maintain indices, are searchable, and provide the structure and protocols needed for individual languages and ecosystems.

- Use of secure hashes of artifacts to verify integrity
- Using and exposing software coordinates
- Providing higher-level cloud/enterprise file/document sharing
- Subsumes the function of container registries
- Constrainable by policy for contractual, regulatory, licensing and engineering concerns

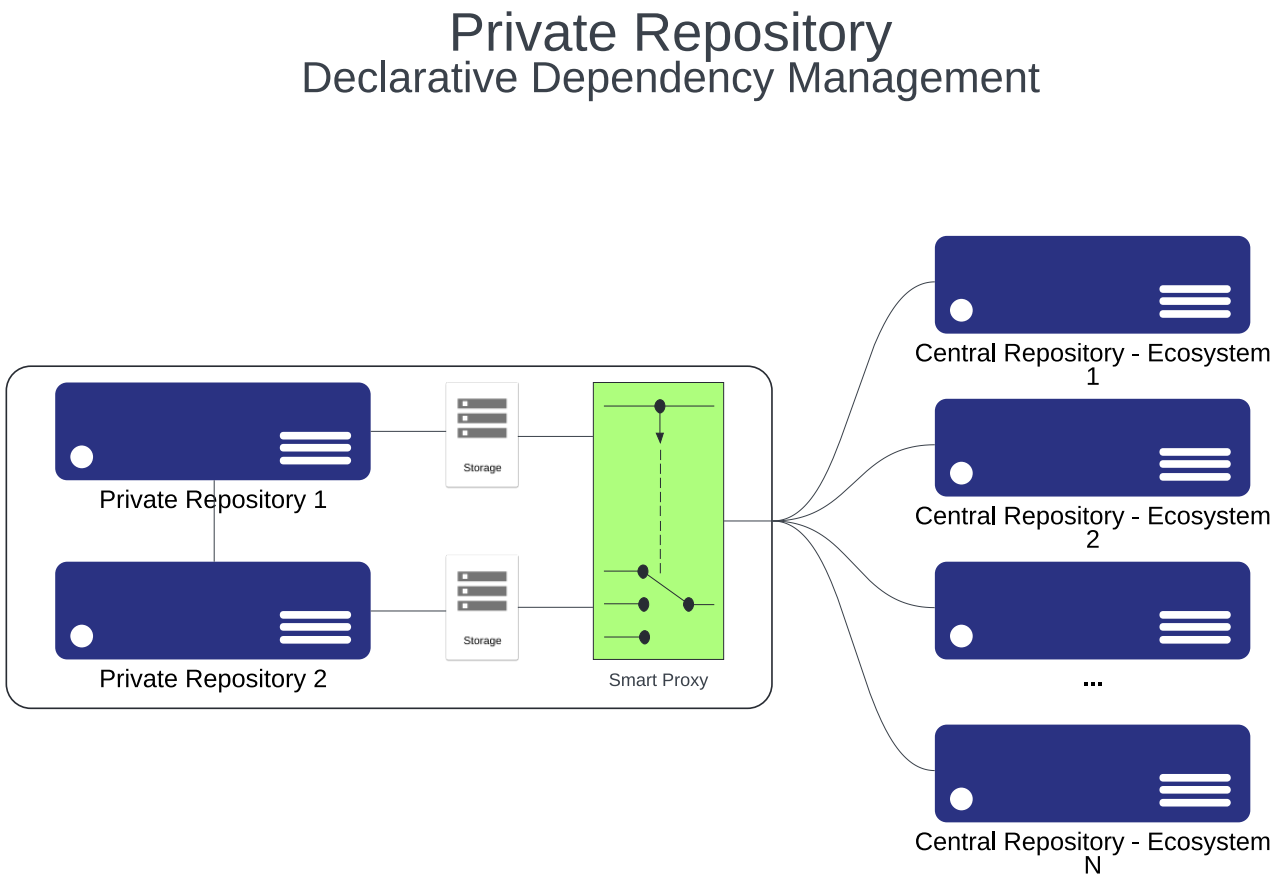

Figure 9.2 Binary Repository as an Archive and Smart Proxy

9.7 Source Repository

The source repository function includes archival of the following artifacts and artifact types and their histories:

- Source code
- Configuration
- License(s) per project
- Declarative dependency enumeration

9.8 Messaging / Event-driven / Asynchronous Architecture

These related architectural elements are grouped because they facilitate similar things: scalable asynchronous processing related to the coordination of distributed development in the automotive supply chain. The introduction of a message broker or messaging infrastructure affords the ability to both generate asynchronous notifications and respond agilely to external events. In the context of SBOM-related activities in the automotive supply chain, these may include the following:

- Notification on a topic/channel that a new build is available
- Notification on a topic/channel that a vulnerability disclosure is available

9.9 Continuous Integration and Continuous Delivery

The CI/CD system performs project builds periodically, but can also perform builds on-demand, or in response to an event - like a project configuration/source code commit (change). The CI/CD system is an invaluable tool for software provenance because of the automation it uses and the rigor it applies:

- CI/CD builds use clean environments to do repeatable builds; they eliminate classes of errors related to mismatches between development and target environments
- CI/CD builds can check dependencies for available updates and conditionally suggest their substitution for older versions
- CI/CD builds can use compilation/interpretation results and automated testing to designate a build as a success or failure
- CI/CD builds can use static analysis to evaluate (successful) project source and configuration compliance
- CI/CD builds can use license analysis to evaluate (successful) project license compliance
- CI/CD builds can apply vulnerability analysis to (successful) build artifacts
- CI/CD builds can apply digital signatures and calculate hashes for (successful) build artifacts
- CI/CD builds can publish build results and promote (successful) build artifacts to project binary repositories
- CI/CD builds can generate notifications alerting concerned parties about project development updates

9.10 Vulnerability Scanner

Vulnerability scanners analyze project artifacts for known vulnerabilities. Vulnerability scanners may examine project configuration including dependencies, project source code, project binaries, and project runtime including profile, behavior and other characteristics, to detect known vulnerabilities.

Vulnerability scanners use one or more sources of vulnerabilities. These sources may include public and private databases cataloguing known vulnerabilities as depicted in Figure 9.3.

9.11 Cryptography Software

Cryptography software helps ensure the integrity of build artifacts by enabling digital signing, digital signature verification, secure hashing, and encryption and decryption. Cryptography software should be well supported, kept up-to-date, and should exhibit crypto-agility by primarily supporting the currently recommended algorithms, hashes, and practices suggested by industry standards and the US[13] National Institute of Standards and Technology (NIST).

[13] United States of America

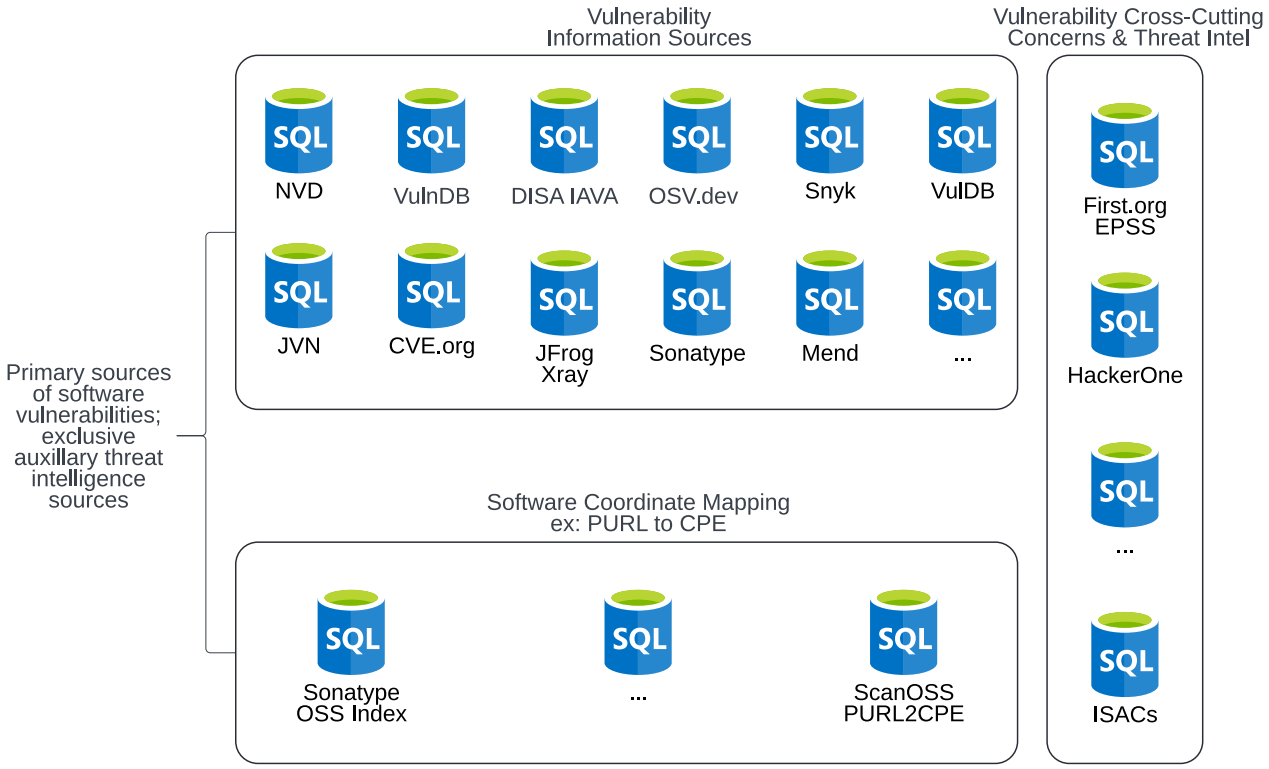

Figure 9.3 Vulnerability Information Sources

9.12 SBOM Generation, Validation, Conversion, Reporting and Inventorying

9.12.1 SBOM Generation

To generate SBOMs, a primary method is to use the tools documented in §B Vendors and Tools for SPDX[14] (§B.2 Software Package Data Exchange Tools) and CycloneDX (§B.1 CycloneDX Tools) from project dependencies. Secondary methods include using SCA[15] tools from §B.4 Software Composition Analysis, including binary analysis and dynamic analysis.

9.12.2 SBOM Sending and Receiving

To send and receive an SBOM, a primary method is to use a binary repository as a shared repository for the sender to publish the SBOM document(s) as build artifacts, and for the receiver to access the SBOM document(s) as build artifacts. Secondary methods to send and receive SBOMs include the use of cloud storage with shared folder(s), email systems, and sending / receiving physical media.

9.12.3 SBOM Validation and Transformation

To validate SBOMs, primary methods include using the tools of §B.2 Software Package Data Exchange Tools and §B.1 CycloneDX Tools. The CycloneDX and SPDX tools of that section can validate SBOMs according to their schemas. To transform SBOMs, primary methods include generation using the methods of SBOM generation, and secondarily using transformation operations.

[14] Software Package Data Exchange
[15] Software Composition Analysis

9.12.4 SBOM Inventorying

To inventory received SBOMs, primary methods include using a software composition analysis platform, such as one of those explored in §B.4 Software Composition Analysis. An alternative method includes the inventorying of SBOMs as product compositions in a business' ERP[16] platform, such as SAP or Oracle ERP. Secondary methods of inventorying received SBOMs include the storage of these artifacts in custom datastores, or document / CMSs[17].

9.12.5 SBOM / Software Product Vulnerability Analysis

To analyze software for vulnerabilities, primary methods include analyzing project SBOMs, or project dependencies for vulnerabilities using vulnerability analysis software explored in §B.3 Vulnerability Scanners. Secondary methods to analyze software for vulnerabilities use SCA software (§B.4 Software Composition Analysis), including binary analysis and dynamic analysis to perform reasoning about software product compositions, and the vulnerability analyses that follow.

9.12.6 SBOM / Software Product Analytics

Primary methods to analyze SBOMs include the use of SCA products discussed in §B.4 Software Composition Analysis, and their analyses. Secondary methods include the use of customized analytical systems designed to add insight to the inventoried software compositions and vulnerabilities. Analytics systems like Tableau from SalesForce, PowerBI from Microsoft, Cognos from IBM, and Hyperion from Oracle typify analytical software packages available in automotive enterprises.

9.12.7 SBOM / Software Product Vulnerability Tracking

Primary methods to track vulnerabilities found in software products includes the integration of SCA products §B.4 Software Composition Analysis to issue trackers. Common issue trackers include Atlassian Jira, ServiceNow, Apache Bugzilla, and GitHub Issues.

9.12.8 SBOM Related Automation

Primary methods to automate the processing of SBOMs include CI/CD, and event-driven architectures and the supporting messaging infrastructures; these are explored in the subtopics of this section. Secondary methods of automating SBOM related processing include workflow and business process automation tools, including BPMN[18] (an Object Management Group (OMG) standard `https://www.omg.org/spec/BPMN/2.0/`) and BPEL[19] (an Organization for the Advancement of Structured Information Standards (OASIS) standard `https://www.oasis-open.org/standard/wsbpel/`) workflows. In some cases, enterprise or cloud tooling for issue tracking offers workflow functionality appropriate for SBOM automation. Tertiary methods include traditional daemon, scripting, and other automation facilities.

9.13 Testing Software

Testing is an integral part of software development. As indicated in ISO 26262 [1] and the V-model that standard established, testing and verification are critical to ensuring transparency and traceability to requirements. The SBOM-WG recommends using automated testing software to additionally verify and validate SBOM build artifacts and to do round-trip engineering. That is, to verify that the mapping of software components and dependencies to an SBOM is consistent, and that the mapping of

[16] Enterprise Resource Planning
[17] Content Management Systems
[18] Business Process Model and Notation
[19] Business Process Execution Language

the SBOM back to software components and their dependencies results in consistent software composition graphs. The SBOM-WG recommends memorializing testing software and configuration in software test SBOMs.

9.14 Fuzz Testing (Fuzzing)

9.14.1 Background

Fuzz testing, or fuzzing is a testing technique that can find previously unknown software bugs, including vulnerabilities. Many software bugs including vulnerabilities found in OSS[20] are discovered by the fuzzing efforts of some of the largest technology companies. Their fuzzing targets generally do not include proprietary software, and are not focused on automotive software. The computationally intensive technique uses permutation on program state by varying Application Programming Interface (API) parameter generation and usage to discover API parameters that cause undesirable application behavior. The undesirable behavior, the API parameters used, and the stack traces generated as a result can be used to classify the errors and guide the troubleshooting and resolution efforts. Large-scale fuzz testing is often done in parallel and using information from the runtime profile (coverage-guided fuzzing) to trim the time and computational resources needed respectively for the technique.

9.14.2 Application to Automotive

The automotive industry benefits from technology companies' OSS fuzz testing efforts by incorporation of the issue fixes in the supply chain's upstream. Although helpful for software quality, these efforts and fixes alone are insufficient for software code coverage in automotive products. By the distributed nature of automotive supply chains, the complexity of the software, the heterogeneity of the software and architecture, and the common lack of source code availability, fuzzing has large importance in automotive software quality. Automotive software quality, to include safety necessitates rigorous identification of hitherto unknown software bugs including vulnerabilities. Fuzz testing can improve SBOMs by indicating improved component and component version selections. Moreover, SBOMs can help guide fuzz testing efforts by using the software component lists, origins, and relationships to identify prime fuzz testing targets.

[20] Open Source Software

Section 10

Appendices

A SBOM Examples

The following section has real life examples of the same SBOM in SPDX and CycloneDX json notation. Please note that while they are examples from real software code, these are intended for illustration only.

A.1 Original Equipment Manufacturer SBOM SPDX


```
{
  "SPDXID": "SPDXRef-DOCUMENT",
  "spdxVersion": "SPDX-2.3",
  "creationInfo": {
    "created": "2024-10-18T07:43:50.450Z",
    "creators": ["Organization: UnifiedMotors", "Tool:
        ExcelSbomConverterCPP-0.3.0"]
  },
  "name": "UnifiedMotorsQ SW: 123",
  "dataLicense": "CC0-1.0",
  "documentNamespace":
      "http://www.unifiedmotorsq.com/spdxdocs/4b6e6c01e992a7b3eb153557eb54263f",
  "packages": [
    {
      "SPDXID": "SPDXRef-Package-b87e",
      "name": "acme_module_q",
      "versionInfo": "890023-e5",
      "supplier": "Organization: acme_battery",
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:acme_battery:acme_module_q:890023-e5:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    },
    {
      "SPDXID": "SPDXRef-Package-d3ff",
      "name": "genmodxe",
      "versionInfo": "6.7",
      "supplier": "Organization: unified_motors",
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:unified_motors:genmodxe:6.7:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    },
    {
      "SPDXID": "SPDXRef-Package-b81a",
      "name": "carconfigurator",
      "versionInfo": "14-280.5",
      "supplier": "Organization: unified_motors",
```

```
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:unified_motors:carconfigurator:14-280.5:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    },
    {
      "SPDXID": "SPDXRef-Package-4e15",
      "name": "bosch_module_x",
      "versionInfo": "a14-20x-1.29.3708",
      "supplier": "Organization: bosch",
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:bosch:bosch_module_x:a14-20x-1.29.3708:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    }
  ],
  "relationships": [
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-b87e"
    },
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-4e15"
    },
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-d3ff"
    },
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-b81a"
    }
  ]
}
```

A.2 Tier-1 SBOM SPDX


```
{
  "SPDXID": "SPDXRef-DOCUMENT",
  "spdxVersion": "SPDX-2.3",
  "creationInfo": {
    "created": "2024-10-18T15:17:35.317Z",
    "creators": ["Organization: Robert Bosch GmbH", "Tool:
        ExcelSbomConverterCPP-0.3.0"]
  },
  "name": "UnifiedMotorsQ SW: 123",
  "dataLicense": "CC0-1.0",
  "documentNamespace":
      "http://www.bosch.com/spdxdocs/7cd8449321f6c622053f20cc76aeb879",
  "packages": [
    {
      "SPDXID": "SPDXRef-Package-3ad7",
      "name": "modulex",
      "versionInfo": "a14-20x-1.29.3708",
      "supplier": "Organization: bosch",
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:bosch:modulex:a14-20x-1.29.3708:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    },
    {
      "SPDXID": "SPDXRef-Package-8de1",
      "name": "linux_bsp",
      "versionInfo": "5.10.52-2.1.2",
      "supplier": "Organization: nxp",
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:nxp:linux_bsp:5.10.52-2.1.2:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    },
    {
      "SPDXID": "SPDXRef-Package-dc8a",
      "name": "qnx7",
      "versionInfo": "660-33.28.1",
      "supplier": "Organization: blackberry",
```

```
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:blackberry:qnx7:660-33.28.1:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    },
    {
      "SPDXID": "SPDXRef-Package-7b31",
      "name": "detroitsw",
      "versionInfo": "1.37",
      "supplier": "Organization: defunct",
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator": "cpe:2.3:a:defunct:detroitsw:1.37:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    }
  ],
  "relationships": [
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-7b31"
    },
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-3ad7"
    },
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-dc8a"
    },
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-8de1"
    }
  ]
}
```

A.3 Tier-2 SBOM SPDX


```
{
  "SPDXID": "SPDXRef-DOCUMENT",
  "spdxVersion": "SPDX-2.3",
  "creationInfo": {
    "created": "2024-10-18T15:22:28.652Z",
    "creators": ["Organization: NXP Semiconductors N.V.", "Tool:
        ExcelSbomConverterCPP-0.3.0"]
  },
  "name": "Linux BSP SW: 8.12",
  "dataLicense": "CC0-1.0",
  "documentNamespace":
      "http://www.nxp.com/spdxdocs/c24f1efbde6f696e27399faa763612fe",
  "packages": [
    {
      "SPDXID": "SPDXRef-Package-3867",
      "name": "linux_bsp",
      "versionInfo": "5.10.52.2.1.0",
      "supplier": "Organization: nxp",
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:nxp:linux_bsp:5.10.52.2.1.0:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    },
    {
      "SPDXID": "SPDXRef-Package-d7da",
      "name": "mwiffiex",
      "versionInfo": "5.10.52.2.1.0",
      "supplier": "Organization: nxp",
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:nxp:mwiffiex:5.10.52.2.1.0:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    },
    {
      "SPDXID": "SPDXRef-Package-4672",
      "name": "linux-imx",
      "versionInfo": "5.10.52.2.1.0",
      "supplier": "Organization: nxp",
```

```
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:nxp:linux-imx:5.10.52.2.1.0:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    },
    {
      "SPDXID": "SPDXRef-Package-e12e",
      "name": "linux_kernel",
      "versionInfo": "5.12.52",
      "supplier": "Organization: linux",
      "downloadLocation": "None",
      "externalRefs": [
        {
          "referenceCategory": "SECURITY",
          "referenceLocator":
              "cpe:2.3:a:linux:linux_kernel:5.12.52:*:*:*:*:*:*:*",
          "referenceType": "cpe23Type"
        }
      ]
    }
  ],
  "relationships": [
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-3867"
    },
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-4672"
    },
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-e12e"
    },
    {
      "spdxElementId": "SPDXRef-DOCUMENT",
      "relationshipType": "DESCRIBES",
      "relatedSpdxElement": "SPDXRef-Package-d7da"
    }
  ]
}
```

A.4 Original Equipment Manufacturer SBOM CycloneDX


```
{
  "bomFormat": "CycloneDX",
  "specVersion": "1.5",
  "serialNumber": "urn:uuid:ea6f796e-8c7f-daaf-d9e3-edfa277bf69c",
  "version": 3,
  "metadata": {
    "timestamp": "2024-08-07T14:40:45.969Z",
    "component": {
      "cpe": "cpe:2.3:a:unified_motors:CarConfigurator:14-280.5:*:*:*:*:*:*:*",
      "name": "CarConfigurator",
      "publisher": "unified_motors",
      "type": "application",
      "version": "6.7"
    },
    "supplier": {
      "name": "Unified Motors",
      "url": [
        "https://www.unified-motors.com"
      ]
    }
  },
  "components": [
    {
      "cpe": "cpe:2.3:a:bosch:bosch_module_x:a14-20x-1.29.3708:*:*:*:*:*:*:*",
      "name": "bosch_module_x",
      "publisher": "bosch",
      "type": "application",
      "version": "a14-20x-1.29.3708"
    },
    {
      "cpe": "cpe:2.3:a:acme_battery:acme_module_q:890023-e5:*:*:*:*:*:*:*",
      "name": "acme_module_q",
      "publisher": "acme_battery",
      "type": "application",
      "version": "890023-e5"
    },
    {
      "cpe": "cpe:2.3:a:unified_motors:genmodxe:14-280.5:*:*:*:*:*:*:*",
      "name": "GenModXE",
      "publisher": "unified_motors",
      "type": "application",
      "version": "14-280.5"
    }
  ]
}
```

A.5 Tier-1 SBOM CycloneDX


```
{
  "bomFormat": "CycloneDX",
  "specVersion": "1.5",
  "serialNumber": "urn:uuid:6e14e9c8-1d81-97ff-ca42-a264db2eeea7",
  "version": 3,
  "metadata": {
    "timestamp": "2024-08-07T15:34:13.437Z",
    "component": {
      "cpe": "cpe:2.3:a:bosch:modulex:a14-20x-1.29.3708:*:*:*:*:*:*:*",
      "name": "modulex",
      "publisher": "bosch",
      "type": "application",
      "version": "a14-20x-1.29.3708"
    },
    "supplier": {
      "name": "Robert Bosch GmbH",
      "url": [
        "https://www.bosch.com"
      ]
    }
  },
  "components": [
    {
      "cpe": "cpe:2.3:a:blackberry:qnx7:660-33.28.1:*:*:*:*:*:*:*",
      "name": "qnx7",
      "publisher": "blackberry",
      "type": "application",
      "version": "660-33.28.1"
    },
    {
      "cpe": "cpe:2.3:a:nxp:linux_bsp:5.10.52-2.1.2:*:*:*:*:*:*:*",
      "name": "linux_bsp",
      "publisher": "nxp",
      "type": "application",
      "version": "5.10.52-2.1.2"
    },
    {
      "cpe": "cpe:2.3:a:defunct:detroitsw:1.37:*:*:*:*:*:*:*",
      "name": "detroitsw",
      "publisher": "defunct",
      "type": "application",
      "version": "1.37"
    }
  ]
}
```

A.6 Tier-2 SBOM CycloneDX


```
{
  "bomFormat": "CycloneDX",
  "specVersion": "1.5",
  "serialNumber": "urn:uuid:71293171-121e-2c12-b610-91d605e84524",
  "version": 3,
  "metadata": {
    "timestamp": "2024-08-07T15:44:39.490Z",
    "component": {
      "cpe": "cpe:2.3:a:nxp:linux_bsp:5.10.52.2.1.0:*:*:*:*:*:*:*",
      "name": "linux_bsp",
      "publisher": "nxp",
      "type": "application",
      "version": "5.10.52.2.1.0"
    },
    "supplier": {
      "name": "NXP Semiconductors N.V.",
      "url": [
        "https://www.nxp.com"
      ]
    }
  },
  "components": [
    {
      "cpe": "cpe:2.3:a:nxp:linux-imx:5.10.52.2.1.0:*:*:*:*:*:*:*",
      "name": "linux-imx",
      "publisher": "nxp",
      "type": "application",
      "version": "5.10.52.2.1.0"
    },
    {
      "cpe": "cpe:2.3:a:linux:linux_kernel:5.12.52:*:*:*:*:*:*:*",
      "name": "linux_kernel",
      "publisher": "linux",
      "type": "application",
      "version": "5.12.52"
    },
    {
      "cpe": "cpe:2.3:a:nxp:mwiffiex:5.10.52.2.1.0:*:*:*:*:*:*:*",
      "name": "mwiffiex",
      "publisher": "nxp",
      "type": "application",
      "version": "5.10.52.2.1.0"
    }
  ]
}
```

B Vendors and Tools

B.1 CycloneDX Tools

Commercial and Opensource

- Auto-ISAC: “Tooling Ecosystem working with CycloneDX”
- Auto-ISAC: “SBoM_Tools”
- https://cyclonedx.org/tool-center/

B.2 Software Package Data Exchange Tools

Commercial and Opensource

- Auto-ISAC: “Tooling Ecosystem working with SPDX”
- Auto-ISAC: “SBoM_Tools”
- https://spdx.dev/use/spdx-tools/

B.3 Vulnerability Scanners

Commercial

- GitHub Security
  - https://docs.github.com/en/code-security
  - https://docs.github.com/en/code-security/supply-chain-security
  - https://docs.github.com/en/code-security/security-advisories
  - https://docs.github.com/en/code-security/dependabot
  - https://docs.github.com/en/code-security/code-scanning
  - https://docs.github.com/en/code-security/secret-scanning
- JFrog xray
  - https://jfrog.com/xray/
- Mend renovate
  - https://www.mend.io/renovate/
- Snyk
  - https://snyk.io/product/snyk-code/

Opensource

- Anchore grype
  - https://github.com/anchore/grype
- CVE-Search Project cve-search
  - https://www.cve-search.org/
  - https://github.com/cve-search/cve-search
  - https://openhub.net/p/cve-search
- Intel cve-bin-tool

  - https://cve-bin-tool.readthedocs.io/en/latest/
  - https://github.com/intel/cve-bin-tool
- Intel cve-check-tool
  - https://github.com/clearlinux/cve-check-tool
  - https://openhub.net/p/cve-check-tool
- OWASP dependency-check
  - https://owasp.org/www-project-dependency-check/
  - https://github.com/jeremylong/DependencyCheck
  - https://openhub.net/p/dependencycheck

B.4 Software Composition Analysis

Commercial

The following vendors presented their commercial SCA[1] products to the SBOM-WG[2]:

- aDolus (now part of Exiger)
  - FACT - Framework for Analysis and Coordinated Trust
  - https://www.exiger.com/perspectives/exiger-acquires-adolus/
- Argus (now Plaxidity)
  - ArgusVVM - Vehicle Vulnerability Management
  - Vehicle Asset Management
  - Vulnerability Detection
  - Assessment and Response
  - https://plaxidityx.com/
- BlackBerry
  - BlackBerry Jarvis
  - SCA
  - SAST[3]
  - https://www.blackberry.com/us/en
- Ceritas
  - Ceritas is a Hardware Bill of Materials (HBOM) vulnerability analysis and remediation platform for critical infrastructure
  - https://ceritas.ai
- Cybeats
  - SBOMStudio
  - https://www.cybeats.com
- Cybellum

[1] Software Composition Analysis
[2] Auto-ISAC SBOM Working Group
[3] Static Application Security Testing

  - The Cybellum Product Security Platform
  - https://cybellum.com

- Finite State
  - Finite State Next Generation Platform
  - https://finitestate.io

- Karamba
  - VCode Automotive Binary Analysis
  - Used during software validation, VCode helps the Original Equipment Manufacturers (OEMs) and the Tier-1s to automatically identify supply-chain cybersecurity issues and address them
  - https://www.karambasecurity.com

- Manifest
  - Manifest
  - The end-to-end SBOM management platform
  - https://www.manifestcyber.com

- Netrise
  - The NetRise Platform is a comprehensive binary analysis platform
  - https://www.netrise.io

Opensource

- OWASP Dependency-Track
  - https://owasp.org/www-project-dependency-track/
  - https://dependencytrack.org
  - https://github.com/DependencyTrack
  - https://openhub.net/p/dependency-track

B.5 Identity and Access Management

Commercial

- Amazon IAM
- Google IAM
- IBM IAM
- Microsoft IAM
- Okta / Auth0
- Oracle IAM
- Ping

Opensource

- Apache Syncope
  - https://syncope.apache.org
  - https://github.com/apache/syncope
  - https://openhub.net/p/syncope
- Casdoor
  - https://casdoor.org/
  - https://github.com/casdoor/casdoor
- Evolveum MidPoint
  - https://evolveum.com/midpoint/
  - https://github.com/Evolveum/midpoint
  - https://openhub.net/p/midPoint
- FusionAuth
  - https://fusionauth.io
  - https://github.com/fusionauth
- Janssen (commercially - Gluu Flex https://gluu.org/flex/)
  - https://github.com/JanssenProject/
  - https://github.com/JanssenProject/jans
  - https://openhub.net/p/janssen
- Keycloak
  - https://www.keycloak.org
  - https://github.com/keycloak/keycloak
  - https://openhub.net/p/keycloak
- Mitreid Connect
  - http://mitreid-connect.github.io/
  - https://github.com/mitreid-connect/
- OpenIAM
  - https://www.openiam.com/
  - https://github.com/OpenIdentityPlatform
  - https://openhub.net/p/openiam-idm-ce
- Ory
  - https://www.ory.sh
  - https://github.com/ory
  - https://openhub.net/p/ory-hydra
- WSO2 Identity
  - https://wso2.com/identity-server/

  - https://github.com/wso2
  - https://github.com/wso2/product-is
  - https://openhub.net/p/wso2-identity-server

B.6 Cryptography Software

Commercial

(Empty)

Opensource

- Awesome Cryptography (List)
  - https://github.com/sobolevn/awesome-cryptography
- Bouncy Castle
  - https://www.bouncycastle.org
  - https://github.com/bcgit/
  - https://openhub.net/p/p_5523
- Crypto++
  - https://github.com/weidai11/cryptopp
  - https://openhub.net/p/cryptopp
- Libressl
  - http://www.libressl.org
  - https://github.com/libressl
  - https://openhub.net/p/libressl
- OpenSSL
  - https://www.openssl.org/
  - https://github.com/openssl/openssl
  - https://openhub.net/p/openssl

  Note: These TLS/Crypto suites are incomplete cryptographic software packages for SBOM purposes.
- gnuTLS
  - https://gnutls.org
- GPG
  - https://www.gnupg.org
- mbedTLS
  - https://www.trustedfirmware.org/projects/mbed-tls/
- WolfSSL
  - https://www.wolfssl.com

B.7 Messaging Platforms and Software

Commercial

- AWS
  - MQ
    - https://aws.amazon.com/amazon-mq/
  - SNS
    - https://docs.aws.amazon.com/sns/
  - SQS
    - https://aws.amazon.com/sqs/
- Google
  - Pub/Sub
    - https://cloud.google.com/pubsub?hl=en
- MS Azure
  - Event Grid
    - https://azure.microsoft.com/en-us/products/event-grid/
  - Event Hubs
    - https://azure.microsoft.com/en-in/products/event-hubs/
  - Web PubSub
    - https://azure.microsoft.com/en-us/products/web-pubsub/

Opensource

- Apache ActiveMQ
  - https://activemq.apache.org
  - https://openhub.net/p/activemq
- Apache Qpid
  - https://qpid.apache.org
  - https://openhub.net/p/qpid-cpp-broker
- Blazing MQ
  - https://bloomberg.github.io/blazingmq/
  - https://github.com/bloomberg/blazingmq
- Kafka
  - https://kafka.apache.org
  - https://openhub.net/p/apache-kafka
- RabbitMQ

  - https://www.rabbitmq.com
  - https://openhub.net/p/rabbitmq

- Redis + Redis Queue
  - https://redis.io
  - https://redis.io/glossary/redis-queue/
  - https://github.com/redis/redis
  - https://github.com/rq/rq
  - https://openhub.net/p/redis

B.8 Binary Repository

Commercial

- Azure Artifacts
  - https://azure.microsoft.com/en-us/products/devops/artifacts/
- Cloudsmith
  - https://cloudsmith.com/
- GitHub Packages
  - https://github.com/features/packages
- Google Artifact Registry
  - https://cloud.google.com/artifact-registry
- Indeo ProGet
  - https://inedo.com/proget

Opensource

- Artipie
  - https://www.artipie.com
  - https://github.com/artipie/artipie
- JFrog Artifactory OSS
  - https://releases.jfrog.io/artifactory/bintray-artifactory/org/artifactory/oss/jfrog-artifactory-oss/
- Sonatype Nexus Repository OSS
  - https://github.com/sonatype/nexus-public
  - https://help.sonatype.com/en/download.html
  - https://openhub.net/p/nexus-oss

C Languages and Ecosystems

Table 10.1 Languages and Ecosystems

Language[1]	Ecosystem[2]	Memory Safe[3]	Status[4]
Assembly	✗	✗	✗
C / C++	Conan	✗	✗
Objective-C	✗	✗	✗
Carbon	✗	✓[5]	✓
Zig	✗	✓[5]	✓
Rust	Cargo	✓	✓
Python	PyPi	✓	✓
Go	Go Packages	✓	✓
Java / JVM Languages	Maven Central	✓	✓
Typescript / Javascript	npm / yarn	✓	✓
.NET / CIL Languages	NuGet	✓	✓
Swift	Swift Packages	✓	✓

[1] Language - Programming language

[2] Ecosystem - Programming language ecosystem / package repository

- **Conan** `https://conan.io/center`
- **Cargo** `https://crates.io`
- **PyPi** `https://pypi.org`
- **Go Packages** `https://pkg.go.dev`
- **Maven Central** `https://www.maven.org/`
- **npm** / **yarn** `https://www.npmjs.com`
- **NuGet** `https://www.nuget.org`
- **Swift Packages** `https://www.swift.org/packages/`

[3] Memory Safe - Programming language default memory safety paradigm

[4] Status - Programming language support, uptake, popularity, and suitability for new projects

[5] The default memory safety paradigm is planned or incomplete

D References

[1] ISO/TC 22/SC 32. *Road vehicles Functional safety*. 26262. Parts 1-12. ISO, 2018. URL: https://www.iso.org/standard/68383.html (cit. on pp. 14, 21, 64, 91).

[2] ISO/SAE. *Road Vehicles - Cybersecurity Engineering ISO/SAE21434*. 21434. Current; Issued 2021-08-31. ISO/SAE, 2021. URL: https://www.sae.org/standards/content/iso/sae21434/ (cit. on pp. 3, 14, 21, 25–27, 33, 47–50, 88, 91, 93).

[3] Shailesh Y. Rangari. *Why Are Regulations Demanding SBOM Adoption*. https://www.isaca.org/resources/news-and-trends/industry-news/2023/why-are-regulations-demanding-sbom-adoption. Accessed: 2024-09-03. Mar. 2023 (cit. on p. 39).

[4] United Nations Economic Commission for Europe (UNECE). *UNECE World Forum for Harmonization of Vehicle Regulations: Regulation No. 155 (Cybersecurity and Cybersecurity Management System)*. R155. Regulation applies from 22 January 2021. United Nations Economic Commission for Europe (UNECE), 2021. URL: https://unece.org/transport/vehicle-regulations (cit. on pp. 3, 15, 21, 47, 49, 93).

[5] United Nations Economic Commission for Europe (UNECE). *UNECE World Forum for Harmonization of Vehicle Regulations: Regulation No. 156 (Software Update and Software Update Management System)*. R156. Regulation applies from 22 January 2021. United Nations Economic Commission for Europe (UNECE), 2021. URL: https://unece.org/transport/vehicle-regulations (cit. on pp. 3, 15, 93).

E Acronyms

ABS Anti-lock Braking System 19

API Application Programming Interface. See also: F API 65

Auto-ISAC Automotive Information Sharing and Analysis Center v, 3, 4, 10, 14, 41, 42, 47, 50, 55, 57

AUTOSAR AUTomotive Open System ARchitecture 26, 30

BPEL Business Process Execution Language 64

BPG Best Practice Guide 42

BPMN Business Process Model and Notation 64

BSW Basic SoftWare 26

C-SCRM Cybersecurity Supply Chain Risk Management 59

CCPA California Consumer Privacy Act 21

CI/CD Continuous Integration and Continuous Delivery 33, 60, 62, 64

CIA Cybersecurity Interface Agreement 4, 25–27, 41

CISA Cybersecurity and Infrastructure Security Administration 9, 10, 13–16, 39

CMS Content Management System 64

COTS Commercial Off-the-shelf 3

CRA Cybersecurity Resilience Act - European Union 12, 14

CSAF Common Security Advisory Framework 13, 16, 31, 39, 50

CSMS Cybersecurity Management System 47, 49

CVE Common Vulnerabilities and Exposures 51, 56, 57

CVSS Common Vulnerability Scoring System 48

CWE Common Weakness Enumeration 40, 56

DFARS Defense Federal Acquisition Regulation Supplement 11

DHS Department of Homeland Security 9, 15

DIA Development Interface Agreement 25–27

DOC Department of Commerce 10, 11

DoD Department of Defense 11

DoE Department of Energy 12, 13

DOT Department of Transportation 9

ECU Electronic Control Unit. See also: F Electronic Control Unit (ECU) 3, 19, 30, 34

EO Executive Order 9, 10, 39

EOL End of Life. See also: F End-of-Support, End-of-Life 47, 59

ERP Enterprise Resource Planning 64

EU European Union 12, 14

EV Electric Vehicle 12

FAQ Frequently Asked Questions 10

FAR Federal Acquisition Regulation 11

FCC Federal Communications Commission 12

FDA Food and Drug Administration 11, 13

FIDO Fast Identity Online 60

FOSS Free and Open Source Software 56

GA General Availability 34

GDPR General Data Protection Regulation - European Union 21

general-purpose OS general-purpose Operating System 34

H-ISAC Health Information Sharing and Analysis Center 11, 13

HDO Health Delivery Organization 13

IAM Identity and Access Management 27, 59

ICT Information and Communications Technology 14

INL Idaho National Labs 12

IoT Internet of Things 12

IP Intellectual Property 4, 5, 19, 26, 28–30, 41

ISO International Organization for Standardization 14, 15, 33, 41

IVI In-Vehicle Infotainment 34

JSON JavaScript Object Notation 31

LF Linux Foundation 15

LIDAR Light Detection and Ranging 21

MDMA Medical Device Manufacturers Association 13

METI Ministry of Economy, Trade and Industry - Japan 13

MFA Multi-factor Authentication 60

MISRA Motor Industry Software Reliability Association 40

NDA Non-Disclosure Agreement 4, 26

NHTSA National Highway Traffic and Safety Administration 9, 12, 39

NIST National Institute of Standards and Technology. See also: F National Institute of Standards and Technology (NIST) 11, 16, 19, 28, 31, 40, 62

NTIA National Telecommunications and Information Administration. See also: F National Telecommunications and Information Administration (NTIA) 10, 14

NVD National Vulnerability Database. See also: F National Vulnerability Database (NVD) 13, 19

OASIS Organization for the Advancement of Information Standards. See also: F OASIS Open 16, 64

OAuth Open Authorization 60

OEM Original Equipment Manufacturer. See also: F Original Equipment Manufacturer 3, 4, 15, 20, 26, 29, 55, 56, 79

OIDC OpenID Connect 60

OMG Object Management Group. See also: F OMG 64

OSIM Open Supply Chain Information Modeling 16

OSS Open Source Software 3, 39, 48, 65

OSSF Open Source Security Foundation - Linux Foundation 15

OTA Over-the-Air 45

OWASP Open Worldwide Application Security Project 15, 16, 39

PKI Public Key Infrastructure 41

PoC Proof of Concept 12, 13

PPD Presidential Policy Directive 9, 10

PSIRT Product Security Incident Response Team 10

PURL package URL. See also: F PURL 33–35

RASIC Responsible, Accountable, Supporting, Informed, Consulted [2] 26

RFP Request for Proposal 25

RFQ Request for Quotation 25

RSO Reduced Sign-on 60

RTOS Real-time Operating System 34

SAE Society of Automotive Engineers 14

SAML Security Assertion Markup Language 60

SAST Static Application Security Testing 78

SBOM Software Bill of Materials See also: F Software Bill of Materials (SBOM) 3–6, 9–16, 25–31, 33–36, 39–45, 47–52, 55–57, 59, 60, 62–65

SBOM-WG Auto-ISAC SBOM Working Group v, 3–6, 25, 26, 31, 33, 34, 48, 49, 51, 52, 59, 64, 65, 78, 92

SCA Software Composition Analysis 33, 34, 63, 64, 78

SDL Security Development Lifecycle. See also: F Security Development Lifecycle (SDL) 41–44

SDLC Software Development Life Cycle 33

SHA Secure Hash Algorithm 41

SLA Service Level Agreement 29

SMB Small and Midsize Business 5

SPDX Software Package Data Exchange 15, 16, 30, 31, 49, 63

SSO Single Sign-on 60

TEVEES18A Vehicle Cybersecurity Systems Engineering Committee 14

Tier-1 Tier-1 suppliers sell directly to OEMs. 3, 4, 55, 56, 79

Tier-2 Tier-2 suppliers sell to Tier-1 suppliers, etc. 3, 4, 55

Tier-N Tier-N suppliers sell to Tier-(N-1) suppliers, etc. 3

TLP Traffic Light Protocol. See also: F TLP 93

TTP Tactics, Techniques, and Procedures 48

UNECE United Nations Economic Commission for Europe 15

URL Uniform Resource Locator 39, 40

US United States of America 9, 11–15, 28, 31, 62

USG United States Government 9, 11

VDR Vulnerability Disclosure Report 31

VEX Vulnerability Exploitability Exchange 10, 13, 14, 16, 31, 50, 52

F Glossary

API An Application Programming Interface (API) is a particular set of rules and specifications that a software application client can use to communicate with an implementation provider 86

Attribute In SBOMs, a data field specifying a property of an element. In an SBOM, attributes include name, version, supplier, etc.

Author, SBOM Individual or organization responsible for the creation and support of the SBOM for supplied software.

Author, Software Individual or organization responsible for creation and support of supplied software or a component included in supplied software.

Component A library, software toolkit, method, subroutine, function, or similar unit of software incorporated into a product or other supplied software. Components may be open source or proprietary, and may be maintained by the supplier or by outside entities. Components can be large or small, with multiple functions like operating systems or single-function like browser plug-ins. Components may include other components. A component in an SBOM is described by an SBOM element.

Consume/Consumer To receive/to be the receiver of a component, product, or supplied software from a supplier.

Contract, Contractual Agreement An agreement between private parties creating mutual obligations enforceable by law. https://www.law.cornell.edu/wex/contract

CycloneDX CycloneDX provides a general-purpose, machine-readable way to define virtually any type of standard. Security standards such as OWASP ASVS, MASVS, SCVS, and SAMM are available in CycloneDX format. Standards from other bodies are available as well. Additionally, organizations can create internal standards and represent them in CycloneDX. 15, 16, 31, 39, 48, 49, 63

defense in depth Layered cybersecurity defenses. 28

Dependency In software and SBOMs, the relationship of a downstream component to an upstream component, (i.e., if component A is included in component B, B has a dependency on A).

DevOps A software development methodology that accelerates the delivery of higher-quality applications and services by combining and automating the work of software development and IT operations teams. With shared tools and practices, including small but frequent updates, software development becomes more efficient, faster and more reliable. https://www.ibm.com/topics/devops.

DevSecOps An application development practice that automates the integration of security and security practices at every phase of the software development lifecycle, from initial design through integration, testing, delivery and deployment. https://www.ibm.com/topics/devsecops.

Downstream In a software supply chain, components that use, are built upon, or have a dependency upon, another component. Also, the relationship of a consumer to a supplier. Downstream components use functionality from an upstream component. In an SBOM, the ultimate downstream component is the supplied software itself. The Report follows the NTIA convention for usage of the terms downstream and upstream.

Electronic Control Unit (ECU) Automotive engineering term for a computing unit incorporating one or more microprocessors and/or microcontrollers. An ECU executes instructions from embedded software to perform functions in vehicle systems and operations. 86

Element A line-item in an SBOM defining a component as it is used in supplied software. Elements include the identifiers, attributes, and relationships of an included component.

End-of-Support, End-of-Life A point in time after which the original supplier or other service suppliers no longer provide parts, service, and/or updates for the software release described by the SBOM. The terms are sometimes ambiguous and may require explicit definitions by suppliers and customers. 87

Hierarchy In an SBOM, the tree-like structure in which the supplied software is composed of typically smaller components, which may themselves be composed of other components, etc.

Incident A discovery and/or execution of an exploit that compromises cybersecurity; i.e. when a threat actually occurs in the real-world.

Includes In an SBOM, the relationship between a downstream component and an upstream component which it uses. For example, Apache Struts includes the upstream log4j library because Struts depends on log4j library presence at runtime.

ISO 26262 [1] International Organization for Standardization (ISO) Road vehicles - Functional safety 14, 21, 64

ISO/SAE 21434 [2] International Organization for Standardization (ISO) / SAE International Surface Vehicle Standard for, Road Vehicles - Cybersecurity Engineering, # 21434. Supersedes and cancels SAE J3061_201601. 3, 14, 21, 25–27, 33, 47–50

Licensing Terms Any legal requirements for use of a given piece of software, including formal contracts, open source required agreements, click-through end-user license agreements (EULA), etc.

Must Term meaning “obligatory” and defined by several standards and practices. “Must” is used in the Report only to quote or paraphrase a referenced document (e.g., UNECE WP.29). All Report recommendations are optional and not mandatory, and therefore use the term “should.”

National Institute of Standards and Technology (NIST) U.S. Department of Commerce. https://www.nist.gov. 88

National Telecommunications and Information Administration (NTIA) U.S. Department of Commerce. https://www.ntia.gov. The NTIA’s archives on SBOMs can be found at https://www.ntia.gov/page/software-bill-materials. 88

National Vulnerability Database (NVD) administered by MITRE Corp. on behalf of NIST 88

OASIS Open Organization for the Advancement of Structured Information Standards (OASIS). https://www.oasis-open.org/. 88

OMG Object Management Group (OMG). https://www.omg.org/. 88

Original Equipment Manufacturer In the context of the Report, a vehicle manufacturer. 88

Product Supplied software which is packaged and distributed by a software vendor. In an SBOM, most often used to refer to the final software and SBOM provided to an end-user customer such as an OEM, inclusive of all components.

provenance The chronology of the origin, development, ownership, location, and changes to a system or system component and associated data. It may also include personnel and processes used to interact with or make modifications to the system, component, or associated data. 28, 34, 36, 40, 59, 62

PURL Package URL. A purl is a URL string used to identify and locate a software package in a mostly universal and uniform way across programming languages, package managers, packaging conventions, tools, APIs and databases. https://github.com/package-url/purl-spec 88

Recommended Desirable, but not prescriptive or mandatory.

Report The "Software Bill of Materials Information Report" provides recommendations based on the studies, exercises, and discussions of the SBOM-WG v, 5, 41

Security Development Lifecycle (SDL) A description of the phases software goes through during its development and deployment with particular focus on the cybersecurity considerations for each phase. Often part of a defined process which creates an expectation of security-related activities performed, and the artifacts or work-products created. 89

Should This Guide adopts the IEEE 29148-2018-11 definition that desired preferences or goals, or non-mandatory and non-binding provisions, use the word "should" rather than alternatives such as "must" or "will."

Software Bill of Materials (SBOM) A formal, machine-readable inventory of software components and dependencies, information about those components, and their hierarchical relationships. An SBOM is a subset of a general Bill of Materials, focusing on software elements of a specific software release, for use in supply chain management. The Software Bill of Materials (SBOM) accompanies the software release and includes a minimum set of attributes for the purpose of aiding in traceability, risk management, vulnerability tracking and version control. The SBOM does not change unless the software release it represents changes. Additional attributes can be accounted for in a software attributes list separately from an SBOM, which can change without affecting the SBOM associated with the software release and is also considered in this documentation. 89

Supplied Software A specific version of a software release, described by an accompanying SBOM. May be embedded in and distributed as part of an automotive part or system. As used in the Report, similar to product but meant to mean any software package or component that can be consumed whether packaged as a commercial offering or otherwise.

Supplier (of software) Entity that creates or is responsible for an identifiable software component, product, or supplied software which is described by an SBOM. Opposite of consumer.

Threat In in-vehicle cybersecurity, any circumstance or event with the potential to adversely impact vehicle systems via unauthorized access, destruction, disclosure or modification of information, denial of service, disablement, causing control malfunction, etc.

Threat Analysis and Risk Assessment (TARA) A formal method used to determine the susceptibility of a system to cyberattacks, and to prioritize the nature of the threat.

tier Links in a supply chain that separate a supplier from the OEM. Tier-1 suppliers sell directly to OEMs, Tier-2 suppliers sell to Tier-1 suppliers, etc. 5, 20

TLP Traffic Light Protocol. A set of designations used to ensure that sensitive information is shared with the appropriate audience. It employs four colors to indicate expected sharing boundaries to be applied by the recipient(s). https://www.cisa.gov/news-events/news/traffic-light-protocol-tlp-definitions-and-usage 89

TLP:AMBER Sources may use Traffic Light Protocol (TLP):AMBER when information requires support to be effectively acted upon, yet carries risk to privacy, reputation, or operations if shared outside of the organizations involved. Note that TLP:AMBER+STRICT should be used to restrict sharing to the recipient organization only. 4, 5

UNECE WP.29 / R155 [4] United Nations Economic Commission for Europe (UNECE), Working Party (WP), Proposals for Interpretation Documents for UN Regulation, No. 155 (Cyber security and cyber security management system). 3, 15, 21, 47, 49

UNECE WP.29 / R156 [5] United Nations Economic Commission for Europe (UNECE), Working Party (WP), Proposals for Interpretation Documents for UN Regulation, No. 156 (Software update and software update management system). 3, 15

Upstream In a software supply chain, components that are included into another component of the supplied software. Upstream components contribute functionality to a downstream component. The ultimate upstream component for any branch of the hierarchy is one that includes no other components, and therefore has a single element SBOM naming only itself. The Report follows the NTIA convention for usage of the terms downstream and upstream.

Vulnerability Weakness or flaw in a system that can be exploited by a threat.

Vulnerability Management Comprehensive management of a system with respect to security vulnerabilities. Should include continuous monitoring of software deficiencies and newly occurring potential security exploits, and the process of fixing and updating these issues.

Work Product [2] The results of cybersecurity activities that fulfill one or more associated requirements. 27, 49