Software Bill of Materials (sBOMs)

(Removing Barriers to the application of tooling to C-SCRM and Software Assurance)

Robert A. Martin
Sr. Secure Software & Technology Prin. Eng.
Trust & Assurance Cyber Technologies Dept.
Cyber Solutions Technical Center

Presented at the DoD, DHS, NIST, and GSA sponsored Software and Supply Chain Assurance Forum hosted at MITRE McLean, VA
Everything is Becoming Software-Enabled and Connected, Either through Task Dependency, Supply Chain, or Information Flow

Today Your System is:
• attackable or
• susceptible to a hazard…

When this Other System gets subverted through:
• an un-patched vulnerability;
• a mis-configuration;
• an application weakness;
• a counterfeit item;
• tainted software or hardware; or
• the system’s susceptibility to a hazard…

We need to be assured that not only are our own systems trustworthy but also everything we depend upon…
The Supply Chain for Software-Enabled Capabilities is Opaque.
Market Transparency through “Software Bill of Materials”

• Third party components are a known systemic risk.
  • Transparency can drive tools and behavior to document risk, support mitigations, and drive better SW development practices.

• NTIA at Commerce launched an open, community-driven, cross-sector “multistakeholder process” to promote software component transparency.
  • Understand the problem and define basics of SBOM
  • Develop use cases across sectors on how such data can be used, today and in the future.
  • Guidance on how to use existing standards to implement SBOM
    • Software ID tags (SWID)
    • Software Package Data Exchange (SPDX)

• Expected draft deliverables late spring 2019

• More info or to join: afriedman@ntia.doc.gov
Use Cases for sBOM

- **Refer-Acquire-Transfer**
  (definition of what it is)

- **Pedigree**
  (history of how it was produced)

- **License Management**
  (conditions about its use)

- **Provenance**
  (chain of custody of it)

- **Integrity**
  (cryptographic basis of unalteredness)

- **Market Transparency**
  (public DBs of components)

- **Formulation**
  (how it was compiled/formed)

- **Assurance**
  (trustworthiness of it)

- **sBOM of a Service**
  (sBOM of sw delivering service)

- **Patch Currency**
  (known fixes are applied to it)

- **Automated Response**
  (sBOM parsing/action)

- **Assured Mkt Transp**
  (public DBs of components)
Provenance and Pedigree

DEFINITIONS

- **Provenance**
  1. The origin, or source of something
  2. The history of ownership of a valued object, or work of art, or literature

- **Pedigree**
  1. A register recording a line of ancestors
  2. An ancestral line: lineage
     The origin and the history of something; broadly: background, history

CONFUSION

- Many use “Provenance” for both meanings.
  The provenance of a piece of data is both the custodianship as well as the lineage of processing and/or derivation that led to the piece of data.

*Definitions (from Merriam-Webster.com)*
Combined Pedigree & Provenance

Provenance (Chain of Custody) of A6 includes Company C and Company B

Pedigree (Lineage) of A6 includes the processes P1 and P2 and other artifacts used to create A6

Separating Pedigree & Provenance

Provenance and Pedigree provide a basis on which to reason about the trustworthiness of an artifact or document

Provenance (Chain of Custody)

Pedigree (Lineage)
Ensuring we have a **verifiable attestation** of the **origin of all code** running in production so that we can have a **root of trust** as we move forward to **defining and enforcing** a collection of **policies** throughout the different stages of the **software development process**.
Code Provenance

What do we get out of all this?

- "Chain of custody" for all code landing in production releases
- Enabling response in the event that anything goes awry
- Flexible, enforced policies for what code is allowed to land in production releases
Use Cases for sBOM

Tools and environments for: Software Composition Analysis Tools

- Web apps
- Cloud apps
- Orchestration of code

Refer-Acquire-Transfer (definition of what it is)
Pedigree (history of how it was produced)
License Management (conditions about its use)
Provenance (chain of custody of it)
Integrity (cryptographic basis of unalteredness)
Market Transparency (public DBs of components)

Contract/Agreement

Developer
Customer

Development Tools

Deployed S/W Tools

Formulation (how it was compiled/formed)
Assurance (trustworthiness of it)
License Management (conditions about its use)
Provenance (chain of custody of it)
Integrity (cryptographic basis of unalteredness)
Market Transparency (public DBs of components)

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 18-1804-45
Potential sBOM elements

- **Between Tiers**
  - **sBOM**
  - **Developer**
  - **Customer**
  - **Contract/Agreement**

- **Spanning Tiers**
  - **Supplier**
  - **Components**
  - **Version**
  - **Created Using**
  - **Created By**
  - **Item Hash/Signature**
  - **sBOM Hash/Signature**

- **Refer-Acquire-Transfer**
  - (definition of what it is)

- **Pedigree**
  - (history of how it was produced)

- **License Management**
  - (conditions about its use)

- **Provenance**
  - (chain of custody of it)

- **Integrity**
  - (cryptographic basis of unalteredness)

- **Market Transparency**
  - (public DBs of components)

- **Formulation**
  - (how it was compiled/formed)

- **Assurance**
  - (trustworthiness of it)

- **sBOM of a Service**
  - (sBOM of sw delivering service)

- **Patch Currency**
  - (known fixes are applied to it)

- **Automated Response**
  - (sBOM parsing/action)

- **Assured Market Transp**
  - (public DBs of components)

- **Development Tools**
  - **Deployed S/W Tools**

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release: Distribution Unlimited. Case No: 18-1804-45
Potential sBOM elements

- Refer-Acquire-Transfer (definition of what it is)
- Pedigree (history of how it was produced)
- License Management (conditions about its use)
- Provenance (chain of custody of it)
- Integrity (cryptographic basis of unalteredness)
- Market Transparency (public DBs of components)

- Supplier
- Components (sources, executables, patches)
- Version
- Created Using
- Created By
- Item Hash/Signature

- Formulation (how it was compiled/formed)
- Assurance (trustworthiness of it)
- sBOM of a Service (sBOM of sw delivering service)
- Patch Currency (known fixes are applied to it)
- Automated Response (sBOM parsing/action)
- Assured Mkt Transp (public DBs of components)

- Development Tools
- Deployed S/W Tools

- Between Tiers
- Spanning Tiers
- sBOM

- Developer
- Customer

- Contract/Agreement
Potential sBOM elements

**Between Tiers**
- **Supplier**
- **Components**
  - (sources, executables, patches)
- **Version**
- **Created Using**
- **Created By**
- **Item Hash/Signature**

**Spanning Tiers**
- sBOM Hash/Signature

**Development Tools**
- **Developer**
- **Contract/Agreement**
- **Customer**
- **Deployed S/W Tools**

**Potential sBOM elements**
- **Refer-Acquire-Transfer**
  - (definition of what it is)
- **Pedigree**
  - (history of how it was produced)
- **License Management**
  - (conditions about its use)
- **Provenance**
  - (chain of custody of it)
- **Integrity**
  - (cryptographic basis of unalteredness)
- **Market Transparency**
  - (public DBs of components)

**Formulation**
- (how it was compiled/formed)

**Assurance**
- (trustworthiness of it)

**sBOM of a Service**
- (sBOM of sw delivering service)

**Patch Currency**
- (known fixes are applied to it)

**Automated Response**
- (sBOM parsing/action)

**Assured Mkt Transp**
- (public DBs of components)
Potential sBOM elements

**Development Tools**
- Refer-Acquire-Transfer (definition of what it is)
- Pedigree (history of how it was produced)
- License Management (conditions about its use)
- Provenance (chain of custody of it)
- Integrity (cryptographic basis of unalteredness)
- Market Transparency (public DBs of components)

**sBOM of a Service**
- Formulation (how it was compiled/formed)
- Assurance (trustworthiness of it)
- sBOM of a Service (sBOM of sw delivering service)
- Patch Currency (known fixes are applied to it)
- Automated Response (sBOM parsing/action)
- Assured Mkt Transp (public DBs of components)

**Contract/Agreement**
- Supplier
- Components (sources, executables, patches)
- Version
- Created Using
- Created By
- Item Hash/Signature

**Between Tiers**
- **Developer**
- **Customer**

**Spanning Tiers**
- sBOM

**Deployed S/W Tools**
<table>
<thead>
<tr>
<th>PARTS</th>
<th>COMPOUND PARTS</th>
<th>FINAL GOODS ASSEMBLED</th>
<th>OPERATOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTERPRISE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MEDICAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FINANCIAL SERVICES</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INDUSTRIAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$OTHER</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
All types of Capabilities are becoming Software-Enabled...

- Medical
- Buildings
- Aeronautics
- Manufacturing
- Energy
- Shipping
- Vehicles
These Changes Go Well beyond Traditional Information Technology...

Water Treatment

Status & Health Monitoring

Oil & Gas

Hydro Power & Dam Mngt

Smart Munitions

Remote Management
Need Secure, Safe, Reliable, and Resilient Behavior that Upholds Privacy Expectations

- IT Risk
  - Loss of information or service
  - Loss of reliability or safety
- Production
  - Loss of life or property

© 2017 Gartner. All rights reserved.
- Need Assurance of More Than Security -

Need Assured Trustworthy Systems
Software Development, Integration, and Management Tools

- Software Composition Analysis Capabilities
- Test
- Operations
- Development "Desktops"

Source Code Repos (Public & Private)

Package Repos (Public & Private)

- Build choreography
- Vulnerability Information
- Licensing Information
Source and Package repos

docker, Unified Agent (File System Agent (FSA)), GitLab, kubernetes, SourceForge, Launchpad, CodePlex, Savannah, CCPForge GitHub, JFrog Artifactory, JFrog Xray, inedo, Amazon ECR, Google Container Registry, Azure Container Registry, Bit Bucket, Subversion, ProjectLocker, CloudForge, Fog Creek Kiln, Codeplane, Assembla, Beanstalk, Codebase

Software Composition Analysis:
Sonatype
Black Duck (Synopsys)
WhiteSource (with plugins)
Protex, Palamida

Developer Desktops (Embedded, Web, Cloud, Desktops/Servers)
IDEs: LINX, NetBeans, Cloud9 IDE, Zend Studio, Atom, Spiralogics
Application Architecture, CodeLobster, CodeCharge Studio, CodePen,
Xcode, Eclipse, Android Studio, Code Blocks, BlueJ, MPLAB

Frameworks: Bootstrap, Expression Studio, HTML5 Builder, Online
Cloud Tools: Kwatee, Azure

Build Choreography
Jenkins, Travis CI, Final builder, CruiseControl, Integrity, GoCD,
Urbancode, Autorabit, CircleCI, Buildkite, TeamCity, Wercker, Bitrise,
Bamboo, Strider, Gitlab CI
The Basics of an Assurance Case

Claim = assertion to be proven

Argument = how evidence supports claim

Evidence = required documentation

Claim
  └── Sub-Claim
      └── Argument
          └── Evidence

Sub-Claim
  └── Argument
      └── Evidence

Assumptions & Preconditions
The Assurance Case

Medical
Space
Aeronautics
Rail
Automotive
Shipping
Autonomous
Critical Infrastructure
Cyber Physical Systems...
The Assurance Case for a System Builder using Assured Components

Exchange and Composition of Assurance Cases between tools and programs
The Assurance Case for a System Builder using Assured Components

Exchange and Composition of Assurance Cases between tools and programs
Multiple Sources of Assurance Evidence from Across the Lifecycle of the item(s) needing Assurance.

- CONOPS evaluation
- Red Teaming
- Attack Surface Analysis
- Blue Teaming
- Architecture Analysis
- Design Analysis/Review
- Dynamic Runtime Analysis
- Static Analysis
- Malformed Input Testing (Fuzzing)

Assurance Case

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 18-1804-45
Defining “Software Security”

Software security encompasses what a software development organization does to protect a software product and the associated critical data from vulnerabilities, internal and external threats, critical errors, or misconfigurations that can affect performance or expose data. It comprises both organizational processes and product capabilities.

**Organizational processes** include governance structures, strategies, guidance, and clearly defined procedures that guide the development of software in a manner that identifies and incorporates security objectives throughout a product’s lifecycle, protects the integrity of the development environment, applies resources to incident and vulnerability management, and manages the supply chain that supports the software development project.

**Product security capabilities** are technical aspects of specific software products that are useful in enabling the products to address common security challenges, such as protecting data, preventing unauthorized access or use, tracking incidents and vulnerabilities, and managing unforeseen events.

Both organizational processes and product security capabilities are vital elements of software security.

Framework Basics

The Framework identifies best practices relating to both organizational processes and product capabilities across the entire software lifecycle. It is organized into six columns: Functions, Categories, Subcategories, Diagnostic Statements, Implementation Notes, and Informative References.

**Functions** organize fundamental software security activities at their highest level, consistent with the software lifecycle. The Functions are:

- **SECURE DEVELOPMENT**
  - Secure development addresses security in the phase of software development when a software project is conceived, initiated, developed, and brought to market.

- **SECURE CAPABILITIES**
  - Secure capabilities identify key security characteristics recommended for a software product.

- **SECURE LIFECYCLE**
  - Secure lifecycle addresses considerations for maintaining security in a software product from its development through the end of its life.
### Secure Development

<table>
<thead>
<tr>
<th>Category</th>
<th>Subcategory</th>
<th>Diagnostic Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure Coding (SC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SC.1</td>
<td>Threat modeling and analysis</td>
<td>The software is secure against known vulnerabilities, unsafe, and unsafe libraries.</td>
</tr>
<tr>
<td>SC.1.1</td>
<td>Software avoids, or includes documented mitigations for, known security vulnerabilities in included functions and libraries.</td>
<td></td>
</tr>
<tr>
<td>SC.3</td>
<td>Software is developed and output to mitigate common vulnerabilities in software.</td>
<td></td>
</tr>
<tr>
<td>SC.3.2</td>
<td>Software validates input and output to mitigate common vulnerabilities in software.</td>
<td></td>
</tr>
<tr>
<td>SC.3.3</td>
<td>Software encodes data or uses anti-cross site scripting (XSS) libraries.</td>
<td></td>
</tr>
<tr>
<td>SC.4</td>
<td>Standard software assurance measures are employed in the software architecture and design.</td>
<td></td>
</tr>
<tr>
<td>SC.4.1</td>
<td>The software employs static code analysis (SCA) to identify and prevent counterfeiting and tempering.</td>
<td></td>
</tr>
</tbody>
</table>

### Testing and Verification (TV)

<table>
<thead>
<tr>
<th>Category</th>
<th>Subcategory</th>
<th>Diagnostic Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>TV.1</td>
<td>Analysis and validation of the software are conducted.</td>
<td></td>
</tr>
<tr>
<td>TV.1.1</td>
<td>Analysis is performed to identify any residual risks.</td>
<td></td>
</tr>
<tr>
<td>TV.2</td>
<td>Code review using manual and automated techniques is conducted.</td>
<td></td>
</tr>
<tr>
<td>TV.2.1</td>
<td>Code review ensures that the software meets the requirements of the software development.</td>
<td></td>
</tr>
<tr>
<td>TV.3</td>
<td>A preventative test plan is established that considers testing the functionality and security of all components.</td>
<td></td>
</tr>
<tr>
<td>TV.3.1</td>
<td>A preventative test plan is established that considers testing the functionality and security of all components.</td>
<td></td>
</tr>
<tr>
<td>TV.4</td>
<td>Software security controls are configured and implemented.</td>
<td></td>
</tr>
<tr>
<td>TV.4.1</td>
<td>Software development configurations are validated before release.</td>
<td></td>
</tr>
<tr>
<td>TV.4.2</td>
<td>Software is subjected to penetration testing.</td>
<td></td>
</tr>
</tbody>
</table>

### Process and Documentation (PD)

<table>
<thead>
<tr>
<th>Category</th>
<th>Subcategory</th>
<th>Diagnostic Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PD.1</td>
<td>Secure development processes are documented throughout software development.</td>
<td></td>
</tr>
<tr>
<td>PD.1.1</td>
<td>Security policies and procedures are gathered, evaluated, and documented.</td>
<td></td>
</tr>
<tr>
<td>PD.1.2</td>
<td>Security policies and procedures are gathered, evaluated, and documented.</td>
<td></td>
</tr>
<tr>
<td>PD.1.3</td>
<td>Security policies and procedures are gathered, evaluated, and documented.</td>
<td></td>
</tr>
<tr>
<td>PD.1.4</td>
<td>Security documentation outlining best practices for software use by end-users and developers are maintained electronically.</td>
<td></td>
</tr>
<tr>
<td>PD.1.5</td>
<td>Testing and validation activities, including results, are documented.</td>
<td></td>
</tr>
<tr>
<td>PD.1.6</td>
<td>Software development organizations maintain an updated product security testing plan that changes to elements and configurations.</td>
<td></td>
</tr>
</tbody>
</table>

### Supply Chain (SM)

<table>
<thead>
<tr>
<th>Category</th>
<th>Subcategory</th>
<th>Diagnostic Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>SM.1</td>
<td>Software implementation and maintenance plans are aligned with supply chain management.</td>
<td></td>
</tr>
<tr>
<td>SM.1.1</td>
<td>Approved supplier statements are in place to ensure the integrity, security, and confidentiality of application components.</td>
<td></td>
</tr>
<tr>
<td>SM.2</td>
<td>Approved supplier statements are in place to ensure the integrity, security, and confidentiality of application components.</td>
<td></td>
</tr>
<tr>
<td>SM.3</td>
<td>Security requirements are documented in contracts, policies, and standards for vendors supplying software components.</td>
<td></td>
</tr>
<tr>
<td>SM.4</td>
<td>Security requirements are documented in contracts, policies, and standards for vendors supplying software components.</td>
<td></td>
</tr>
<tr>
<td>SM.5</td>
<td>Supply chain incidents are prevented and logged.</td>
<td></td>
</tr>
</tbody>
</table>

---

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 18-1804-45
<table>
<thead>
<tr>
<th>Category</th>
<th>Subcategory</th>
<th>Diagnostic Statement</th>
<th>Category</th>
<th>Subcategory</th>
<th>Diagnostic Statement</th>
<th>Category</th>
<th>Subcategory</th>
<th>Diagnostic Statement</th>
<th>Category</th>
<th>Subcategory</th>
<th>Diagnostic Statement</th>
<th>Category</th>
<th>Subcategory</th>
<th>Diagnostic Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Secure Capabilities</strong></td>
<td><strong>Support for Identity Management and Authentication (SI)</strong></td>
<td></td>
<td><strong>Secure Capabilities</strong></td>
<td><strong>Encryption (EN)</strong></td>
<td></td>
<td><strong>Secure Capabilities</strong></td>
<td><strong>Authorization and Access Controls (AA)</strong></td>
<td></td>
<td><strong>Secure Capabilities</strong></td>
<td><strong>Error and Exception Handling (EE)</strong></td>
<td></td>
<td><strong>Secure Capabilities</strong></td>
<td><strong>Logging (LO)</strong></td>
<td></td>
</tr>
<tr>
<td>SI.1</td>
<td>The software avoids architectural weaknesses that create risk of authentication failure.</td>
<td></td>
<td>EN.1</td>
<td>Software is developed in accordance with an encryption strategy that defines what data should be encrypted and which encryption mechanisms should be used.</td>
<td></td>
<td>AA.1</td>
<td>Software design reflects the principle of least privilege.</td>
<td></td>
<td>LO.1</td>
<td>Software logs a variety of audit log events.</td>
<td></td>
<td>SI.2</td>
<td>The software supports strong identity management and authentication.</td>
<td></td>
</tr>
<tr>
<td>Category</td>
<td>Subcategory</td>
<td>Diagnostic Statement</td>
<td>Category</td>
<td>Subcategory</td>
<td>Diagnostic Statement</td>
<td>Category</td>
<td>Subcategory</td>
<td>Diagnostic Statement</td>
<td>Category</td>
<td>Subcategory</td>
<td>Diagnostic Statement</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>----------------------</td>
<td>----------</td>
<td>-------------</td>
<td>----------------------</td>
<td>----------</td>
<td>-------------</td>
<td>----------------------</td>
<td>----------</td>
<td>-------------</td>
<td>----------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VM 1.1.5</td>
<td>The vendor maintains an up-to-date vulnerability management plan.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 2.5.1</td>
<td>The vendor monitors security posture and prioritizes remediation.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 3.4</td>
<td>The vendor maintains a system to record and track all reports of potential vulnerabilities.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>CF 1.4</td>
<td>The software is deployed with configurations and configuration guidance that facilitate secure installation and operation.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VN 3</td>
<td>Users are notified of a significant security issue when a remediation is in place for each supported version of the affected product.</td>
<td></td>
</tr>
<tr>
<td>VM 2.3</td>
<td>Vulnerabilities are assigned a severity value based on risk, using a standardized scoring methodology.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 2.4</td>
<td>Remediation and mitigation activities are informed by the severity of the vulnerability.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 3.5</td>
<td>The vendor monitors a system to record and track all reports of potential vulnerabilities.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>CF 1.5</td>
<td>The software configuration settings can be altered to tailor security settings to the operating environment.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VN 3.2</td>
<td>Advisory messages notifying users of security issues include information on affected products, applicable versions, and platforms, a unique identification number, and a brief description of the vulnerability and its potential impact.</td>
<td></td>
</tr>
<tr>
<td>VM 3.1</td>
<td>The vendor establishes a clearly defined and easily accessible process for identifying vulnerabilities.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 3.2</td>
<td>A vendor's malware mechanism provides for secure and confidential communication of sensitive vulnerability information.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 3.6</td>
<td>The vendor monitors, in simple and clear language, its policies for interacting with vulnerability reporters, addressing, at minimum, (1) how the vendor would like to be contacted, (2) options for secure communication, (3) expectations for communication from the vendor regarding the status of a reported vulnerability, (4) desired information regarding a potential vulnerability, (5) issues that are out of scope of the vulnerability disclosure program, (6) how submitted vulnerability reports are tracked, and (7) process for vulnerabilities that cannot be resolved.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VN 4.1</td>
<td>Patches or updates are disseminated to end-users according to the vendor's security policy.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>EL 1</td>
<td>The vendor maintains consistent lifecycle guidance.</td>
<td></td>
</tr>
<tr>
<td>VM 3.3</td>
<td>The vendor maintains a coordinated vulnerability disclosure program.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 3.4</td>
<td>The vendor monitors a system to record and track all reports of potential vulnerabilities.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 3.6</td>
<td>The vendor monitors, in simple and clear language, its policies for interacting with vulnerability reporters, addressing, at minimum, (1) how the vendor would like to be contacted, (2) options for secure communication, (3) expectations for communication from the vendor regarding the status of a reported vulnerability, (4) desired information regarding a potential vulnerability, (5) issues that are out of scope of the vulnerability disclosure program, (6) how submitted vulnerability reports are tracked, and (7) process for vulnerabilities that cannot be resolved.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VN 4.2</td>
<td>Patches or updates are transmitted in a manner that prevents exposure of the software image.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>EL 1.2</td>
<td>The vendor clearly communicates decisions to terminate support for a software product to customers and users, identifying the expected support termination date; the anticipated risk of continued product use beyond the termination of support; possible mitigation actions; and options for technical migration to replacement products.</td>
<td></td>
</tr>
<tr>
<td>VM 3.4</td>
<td>The vendor maintains a system to record and track all reports of potential vulnerabilities.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 3.5</td>
<td>The vendor monitors a system to record and track all reports of potential vulnerabilities.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 4.1</td>
<td>The software configuration settings can be altered to tailor security settings to the operating environment.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VN 4.3</td>
<td>Patches or updates are transmitted in a manner that prevents exposure of the software image.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>EL 1.3</td>
<td>The software is continually monitored to ensure that third-party components have not been compromised.</td>
<td></td>
</tr>
<tr>
<td>VM 3.5</td>
<td>The vendor monitors a system to record and track all reports of potential vulnerabilities.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 3.6</td>
<td>The vendor monitors, in simple and clear language, its policies for interacting with vulnerability reporters, addressing, at minimum, (1) how the vendor would like to be contacted, (2) options for secure communication, (3) expectations for communication from the vendor regarding the status of a reported vulnerability, (4) desired information regarding a potential vulnerability, (5) issues that are out of scope of the vulnerability disclosure program, (6) how submitted vulnerability reports are tracked, and (7) process for vulnerabilities that cannot be resolved.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VM 4.2</td>
<td>Patches or updates are transmitted in a manner that prevents exposure of the software image.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>VN 4.4</td>
<td>Patches or updates are transmitted in a manner that prevents exposure of the software image.</td>
<td><strong>SECURE LIFECYCLE</strong></td>
<td>EL 1.4</td>
<td>The software is continually monitored to ensure that third-party components have not been compromised.</td>
<td></td>
</tr>
</tbody>
</table>

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release: Distribution Unlimited. Case No: 18-1804-45
Transparent Assurance as a Basis for Trust - Future

Hardware
- OEM
- ODM
- Modules
- Chips

Software
- Software Integrator
- Solution Provider
- Development Tools
- Software Stack
  - Framework
  - Container
  - Guest OS
  - Hypervisor
  - Firmware

Services
- SaaS
- PaaS
- IaaS

Assurance Case

Balance of Assurance for Secure Software
IIC Journal of Innovation – September 2018 issue on Trustworthiness
https://www.iiconsortium.org/journal-of-innovation.htm

“Assuring Trustworthiness in an Open Global Market of IIoT Systems via Structured Assurance Cases”