Visibility & Control: Addressing Supply Chain Challenges to Trustworthy Software-Enabled Things

Robert A. Martin  
Sr. Secure Software & Technology Prin. Eng.  
Trust & Assurance Cyber Technologies Dept.  
Cyber Solutions Technical Center
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…
How is Software-Enabled and Connected (aka Cyber) Becoming so Pervasive?

Context: 1976 Chevy Vega

The only software was behind the wheel – no microelectronics.
Critical Functions Migrated into Software & Microelectronics (SW/HW)

Electric throttle valve control
Control/Communication Transitioned from Point-to-Point Wiring to Network-based
Critical Functions Are Migrating into Connected SW/HW

- Smart Catalytic Converter
- Beam Shaping Headlights
- Direct fuel injection
- Active suspension
- Electric throttle valve control
- Brake-by-wire
- Steer-by-wire
- Electrically assisted power steering
- 42-V converter
- Anti-Lock Brakes
- Transmission Control
- Lighting
- Air Conditioning
- Power Locks
- Airbag
- Power Seats
- Power Windows

- Seat Motors
- Telematics
- Door Locks
- Pwr Windows
- Engine Controller
- Ignition
- Fuel Injection
- Transmission
- Emissions
- Diagnostics
- Dashboard
- Anti-Lock Brakes
- Connector
- Diagnosis
- Re-programming
- Prognosis
Multiple Types of Networks Are Appearing

- CAN (Controller Area Network)
- MOST (Media Oriented Systems Transport)
- Ethernet AVB (Audio Visual Bridging)
- Ethernet TSN (Time-Sensitive Networking)
- Collision Detection System
- Brake-by-Wire System
- FlexRay
- LIN (Local Interconnect Network)
- Multifunction Keyless System
Many Critical Functions Now Need to be Updated and Sustained...
The Connectivity and Complexity of Connected Software-Enabled Systems is Still Expanding

Driverless Cars, ADAS, V2V, V2I, Safety
All types of Enterprises are Facing these Same Changes...

Medical

Buildings

Aeronautics

Manufacturing

Energy

Shipping

Vehicles
These Changes Go Well beyond Traditional Information Technology...

Water Treatment

Status & Health Monitoring

Remote Management

Oil & Gas

Hydro Power & Dam Mngt

Smart Munitions
Secure Behavior

MIND THE GAP

Reliable Behavior

Safe Behavior

Resilient Behavior

Privacy Expectations
– Need Assurance of More Than Security –

Need Assured Trustworthy Systems
Pervasiveness of connected SW & SW-enabled capabilities requires supply chain security skills / new awareness of SW risks

IT Risk

Loss of data or capability

Operational Risk

Loss of safety or reliability

Loss of property or lives

Scratch Built Software

Majority of products built with no 3rd Party dependencies

Assembled Software

Use of open source and 3rd party libraries, modules, frameworks, and services

Multi-party software updating/patching

Traditional Computers

Servers, databases

Healthcare

Implantable Medical

Smart Munitions

Desks, office apps

Aeronautics

Smart Manufacturing

Intelligent Vehicles

Laptops, e-mail

Smart Energy

Water Treatment

Intelligent Shipping

Tablets, browsers

Oil & Gas

Hydro Power

Dam Management

Switches, Routers

Microgrids

Smart Cities

Building Management

Software Enabled Everything

Autonomous Systems

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-24
For Software-Enabled IIoT Version Control is Crucial

Tracking details for SW & HW components
- SW & HW Part numbers/names
- SW & HW versions
- Libraries & Frameworks Used
- Tool Chain Used/Flags/Options
- Languages & versions used
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)

• First phase deliverable mid-November 2019

• More info or to join: afriedman@ntia.doc.gov
NTIA Transparency Phase 1 Final Products

Survey of Existing SBOM Formats and Standards

SOFTWARE COMPONENT TRANSPARENCY: HEALTHCARE PROOF OF CONCEPT REPORT
Drafted as part of a process convened by the National Telecommunications and Information Administration

October 1, 2019
Ecosystem of SW Development, Integration, and Management Tools

- Test
- Operations
- Developer "Desktops"
- Build choreography
- Source Code Repos (Public & Private)
- Package Repos (Public & Private)
- Vulnerability Information
- Licensing Information
- Software Composition Analysis Capabilities

- Ecosystem of SW Development, Integration, and Management Tools
- Licensing Information
- Vulnerability Information
- Package Repos (Public & Private)
- Source Code Repos (Public & Private)
- Build choreography
- Developer "Desktops"
- Operations
- Test
**Source Code & Package Repositories**

Amazon ECR, Assembla, Azure Container Registry, Beanstalk, Bitbucket, Codebase, Docker, GitHub, GitLab, Glitch, Google Container Registry, JFrog Artifactory, JFrog Xray, inedo, Kubernetes, Launchpad, Maven, Nexus (Sonatype), Phabricator, ProjectLocker, Repository Hosting, Savannah, SourceForge, SourceRepo, Subversion, and Unfuddle

**Build & Build Choreography Capabilities**

Ansible, Autorabit, Bamboo, Bitrise, Buildkite, Buildroot, CircleCI, CMak,
CruiseControl, Final builder, GCC, Gitlab CI, GoCD, Integrity, Jenkins, Strider CD, TeamCity, Terraform, Travis CI, Urbancode, and Vagrant

**Developer Desktops** (Embedded, Web, Cloud, Desktops/Servers)


Frameworks: .NET, Angular, Ansible, Apache Spark, ASP.NET, Bootstrap, Chef, Cordova, CryEngine, Django, Drupal, Express, Flask, Flutter, Hadoop, HTML5 Builder, Laravel, Node.js, Pandas, Puppet, React Native, React.js, Ruby on Rails, Spring, TensorFlow, Torch/PyTorch, Unity D, Unreal Engine, Visual Online, Vue.js, and Xamarin

Cloud Tools: Azure, AWS CodeBuild, Cloud Foundry, Google Cloud Build, Kwatee, Pivotal, and Red Hat

Software Composition Analysis:

Black Duck Software Composition Analysis (Synopsys), CAST Highlight (CAST Software), Finate State, FlexNet Code Insite (Flexera), Ion Channel, Insignary, SourceClear, Sonatype, Snyk, and WhiteSource
Usage Scenarios for Tool-to-Tool SBoM

- **Refer, Transfer or Purchase**
  (definition of what it is)

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

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

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

- **Proper and Legal**
  (conditions about its use)

- **Known Sw Vulns**
  (known fixes are applied to it)

- **Assurance**
  (safe-secure-resilient)

- **SBoM of a SW Service**
  (SBoM of sw delivering service)

- **Supply Chain Sequence Integrity**

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-24
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 

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)*

---

Separating Provenance and Pedigree

**Provenance**  
Captures *chain of custody* of an Artifact, Document or Record

**Pedigree**  
Captures the *history* of how an Artifact or Document was produced or derived
**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.
The Path to Code Provenance at Uber

April 17, 2019

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

Uber
Usage Scenarios and Tool-to-Tool SBOM candidate elements

**Usages**

1. Refer, Transfer or Purchase (definition of what it is)
2. Pedigree (history of how it was produced)
3. Provenance (chain of custody of it)
4. Integrity (cryptographic basis of unalteredness)
5. Intellectual Property Constraints
6. Known SW Vulns (known fixes are applied to it)
7. Assurance (secure-safe-resilient)
8. SBOM of a SW Service (SBOM of SW delivering service)
9. Supply Chain Sequence Integrity

**SBoM elements**

- Author of SBOM
- SBOM population method
- SBOM Time-Stamp
- Supplier
- Components (sources, executables, patches)
- Version
- Notes
- Licenses
- Created Using
- Created By
- Item Hash/Signature

**Correlated Info**
Usages

1. Refer, Transfer or Purchase (definition of what it is)
2. Pedigree (history of how it was produced)
3. Provenance (chain of custody of it)
4. Integrity (cryptographic basis of unalteredness)
5. Intellectual Property Constraints
6. Known SW Vulns (known fixes are applied to it)
7. Assurance (secure-safe-resilient)
8. SBoM of a SW Service (SBoM of SW delivering service)
9. Supply Chain Sequence Integrity

SBoM elements

- Author of SBoM
- SBoM population method
- SBoM Time-Stamp
- Supplier
- Components (sources, executables, patches)
- Version
- Notes
- Licenses
- Created Using
- Created By
- Item Hash/Signature

Correlated Info

None
Usages

1. Refer, Transfer or Purchase (definition of what it is)
2. Pedigree (history of how it was produced)
3. Provenance (chain of custody of it)
4. Integrity (cryptographic basis of unalteredness)
5. Intellectual Property Constraints
6. Known SW Vulns (known fixes are applied to it)
7. Assurance (secure-safe-resilient)
8. SBoM of a SW Service (SBoM of sw delivering service)
9. Supply Chain Sequence Integrity

SBoM elements

- Author of SBoM
- SBoM population method
- SBoM Time-Stamp
- Supplier
- Components (sources, executables, patches)
- Version
- Notes
- Licenses
- Created Using
- Created By
- Item Hash/Signature

Correlated Info

None
**Usages**

1. Refer, Transfer or Purchase (definition of what it is)
2. Pedigree (history of how it was produced)
3. Provenance (chain of custody of it)
4. Integrity (cryptographic basis of unalteredness)
5. Intellectual Property Constraints
6. Known SW Vulns (known fixes are applied to it)
7. Assurance (secure-safe-resilient)
8. SBoM of a SW Service (SBoM of sw delivering service)
9. Supply Chain Sequence Integrity

---

**SBoM elements**

- Author of SBoM
- SBoM population method
- SBoM Time-Stamp
- Supplier
- Components (sources, executables, patches)
- Version
- Notes
- Licenses
- Created Using
- Created By
- Item Hash/Signature

---

**Correlated Info**

None
Usages

1. Refer, Transfer or Purchase
   (definition of what it is)

2. Pedigree
   (history of how it was produced)

3. Provenance
   (chain of custody of it)

4. Integrity
   (cryptographic basis of unalteredness)

5. Intellectual Property
   Constraints

6. Known SW Vulns
   (known fixes are applied to it)

7. Assurance
   (secure-safe-resilient)

8. SBoM of a SW Service
   (SBoM of sw delivering service)

9. Supply Chain Sequence
   Integrity

SBoM elements

- Author of SBoM
- SBoM population method
- SBoM Time-Stamp
- Supplier
- Components
  (sources, executables, patches)
- Version
- Notes
- Licenses
- Created Using
- Created By
- Item Hash/Signature

Correlated Info

None
Usages

1. Refer, Transfer or Purchase
   (definition of what it is)
2. Pedigree
   (history of how it was produced)
3. Provenance
   (chain of custody of it)
4. Integrity
   (cryptographic basis of unalteredness)
5. Intellectual Property
   Constraints
6. Known SW Vulns
   (known fixes are applied to it)
7. Assurance
   (secure-safe-resilient)
8. SBoM of a SW Service
   (SBoM of sw delivering service)
9. Supply Chain Sequence
   Integrity

SBoM elements

- Author of SBoM
- SBoM population method
- SBoM Time-Stamp
- Supplier
- Components
  (sources, executables, patches)
- Version
- Notes
- Licenses
- Created Using
- Created By
- Item Hash/Signature

Correlated Info

None
Usages

1. Refer, Transfer or Purchase (definition of what it is)
2. Pedigree (history of how it was produced)
3. Provenance (chain of custody of it)
4. Integrity (cryptographic basis of unalteredness)
5. Intellectual Property Constraints
6. Known SW Vulns (known fixes are applied to it)
7. Assurance (secure-safe-resilient)
8. SBoM of a SW Service (SBoM of sw delivering service)
9. Supply Chain Sequence Integrity

SBoM elements

- Author of SBoM
- SBoM population method
- SBoM Time-Stamp
- Supplier
- Components (sources, executables, patches)
- Version
- Notes
- Licenses
- Created Using
- Created By
- Item Hash/Signature

Correlated Info

Notes on exploitability of vulns
Vulnerability Knowledge Bases
Weakness Knowledge Bases
Assessment Results
Design Review
Code Review
Attack Surface Analysis
Static Analysis
Dynamic Analysis
Fuzz Testing
Pen Testing
Blue Teaming
Red Teaming
Organized as an Assurance Case
**Usages**

1. Refer, Transfer or Purchase (definition of what it is)
2. Pedigree (history of how it was produced)
3. Provenance (chain of custody of it)
4. Integrity (cryptographic basis of unalteredness)
5. Intellectual Property Constraints
6. Known SW Vulns (known fixes are applied to it)
7. Assurance (secure-safe-resilient)
8. SBoM of a SW Service (SBoM of SW delivering service)
9. Supply Chain Sequence Integrity

**SBoM elements**

- Author of SBoM
- SBoM population method
- SBoM Time-Stamp
- Supplier
- Components (sources, executables, patches)
- Version
- Notes
- Licenses
- Created Using
- Created By
- Item Hash/Signature

**Correlated Info**

Logging SBOMs of Services Used

**SBoM of a SW Service**

* (SBoM of SW delivering service)
Usages

1. Refer, Transfer or Purchase (definition of what it is)
2. Pedigree (history of how it was produced)
3. Provenance (chain of custody of it)
4. Integrity (cryptographic basis of unalteredness)
5. Intellectual Property Constraints
6. Known SW Vulns (known fixes are applied to it)
7. Assurance (secure-safe-resilient)
8. SBoM of a SW Service (SBoM of sw delivering service)
9. Supply Chain Sequence Integrity

SBoM elements

- Author of SBoM
- SBoM population method
- SBoM Time-Stamp
- Supplier
- Components (sources, executables, patches)
- Version
- Notes
- Licenses
- Created Using
- Created By
- Item Hash/Signature

Correlated Info

Desired sequence of ordered software supply chain steps, and requirements for each step for a specific project of interest
Whitepaper → CISQ → OMG RFC → ISO Std

- Socialize at Mar19 OMG meeting
- Draft SBoM as a Whitepaper in 3-day CISQ SBoM working session at Sep OMG meeting
- Prototype draft format in tool ecosystem, revise and draft RFC based on prototype results
- Co-submit draft RFC w/CISQ to OMG at Dec19 or Mar20 meeting
- Mar20/Jun20 OMG meeting – charter FTF
- Jun20/Sep20 OMG meeting - approve as OMG Standard
- Sep20/Dec20 Fast Track to ISO
Exploitable Weaknesses, Vulnerabilities & Exposures

- **Weakness**: mistake or flaw condition in ICT architecture, design, code, or process that, if left unaddressed, could under the proper conditions contribute to a cyber-enabled capability being vulnerable to exploitation; represents potential source vectors for zero-day exploits -- Common Weakness Enumeration (CWE) https://cwe.mitre.org/

- **Vulnerability**: mistake in software that can be directly used by a hacker to gain access to a system or network; Exposure: configuration issue of a mistake in logic that allows unauthorized access or exploitation – Common Vulnerability and Exposure (CVE) https://cve.mitre.org/

- **Exploit**: take advantage of a weakness (or multiple weaknesses) to achieve a negative technical impact -- attack approaches from the set of known exploits are used in the Common Attack Pattern Enumeration and Classification (CAPEC) https://capec.mitre.org

The existence (even if only theoretical) of an exploit designed to take advantage of a weakness (or multiple weaknesses) and achieve a negative technical impact is what makes a weakness a vulnerability.
Assurance needs to address the Hazards & Attacks that can impact SW-Based Mission Functions

Known Threats & Hazards

Hazard/Attack Activation Patterns (CAPECs)

Weaknesses (CWEs)

Counter Measures - Actions*

Technical Impacts to Mission Capabilities

Mission/Business Impacts

Hazard/Attack

Hazard/Attack

Hazard/Attack

“Counter Measures - Actions” include:
choices about architecture, design, physical decomposition, and operational approaches;
adding/changing security/safety functions, protection schemes, activities & processes;
use of static & dynamic code assessments, dynamic testing, physical testing, and pen testing;
attack surface & fault-tree analysis, architecture and design reviews

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-24
Assurance needs to address the Hazards & Attacks that can impact SW-Based Mission Functions

“Counter Measures - Actions” include:
choices about architecture, design, physical decomposition, and operational approaches;
adding/changing security/safety functions, protection schemes, activities & processes;
use of static & dynamic code assessments, dynamic testing, physical testing, and pen testing;
attack surface & fault-tree analysis, architecture and design reviews

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-24
Utilizing Appropriate Detection Methods to Collect Evidence to Gain Assurance...

Artifacts

Detection Methods

Coverage

Most Important Quality Issues

CVE, CWE, CAPEC, ...

CONOPS
Requirements
Architecture
Design
Process
Code
Binary
Running Binary
Environment of System
Use of Mission Software

Design Review
Code Review
Attack Surface Analysis
Static Analysis Tool A
Static Analysis Tool B
Dynamic Analysis Tool C
Fuzz Testing
Pen Testing
Blue Teaming
Red Teaming

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-24
Multiple Sources of Assurance Evidence from Throughout the Lifecycle of the item(s) needing Assurance.

- CONOPS evaluation
- Red Teaming
- Attack Surface Analysis
- Architecture Analysis
- Design Analysis/Review
- Static Analysis
- Blue Teaming
- Penetration Testing
- Malformed Input Testing (Fuzzing)
- Dynamic Runtime Analysis
- Red Teaming
- Blue Teaming
- Penetration Testing
- Malformed Input Testing (Fuzzing)
- Dynamic Runtime Analysis
- Design Analysis/Review
- Static Analysis
- Architecture Analysis
- Attack Surface Analysis
- CONOPS evaluation
- Red Teaming
Different Perspectives on Assurance of Trust

**Insurer**
- How do I underwrite?

**Operator**
- How do I use this?
- Can I trust it?
- Am I responsible if it makes a mistake?

**Researcher**
- What technology is needed to ensure trust?

**Commander/Manager**
- Can I reliably use in operations?
- What changes operationally?

**Creator**
- How should I design and build?
- Will I be liable for problems?

**Regulator**
- Is it safe?

**Community**
- Do I want this in my backyard?
- Can I count on it?

**Patron**
- Is it safe?
- Should I use it?
- Can I count on it?

**Acquirer**
- How do I express requirements?
- Will it work the way it should?
Need Standards to Drive Consistency in Discussing and Conveying Assurance due to the Sector-2-Sector linkages.
Establishing Assurance - Reducing Uncertainty

While Assurance does not provide additional security services or safeguards, it does serve to reduce the uncertainty associated with vulnerabilities resulting from:
- Bad practices
- Incorrect & inefficient safeguards

The result of System Assurance is justified confidence delivered in the form of an Assurance Case.

Confidence demands objectivity, scientific method and cost-effectiveness.
Assurance Claims with Support of ‘Substantial’ Reasoning

• Claims are assertions put forward for general acceptance.
• The justification for claim based is on some grounds, the “specific facts about a precise situation that clarify and make good for a claim.”
• The basis of the reasoning from the grounds (the facts) to the claim is articulated.
• Toulmin coined the term “warrant” for “substantial argument”.
• These are statements indicating the general ways of argument being applied in a particular case and implicitly relied on and whose trustworthiness is well established.
• The basis of the warrant might be questioned, so “backing” for the warrant may be introduced. Backing might be the validation of the scientific and engineering laws used.
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
Infusion Pumps Total Product Life Cycle

Guidance for Industry and FDA Staff

Document issued on December 3, 2014
The draft of this document was issued on April 23, 2010.

This document superseded the "Guidance on the Control of Premarket Notification [510(k)] Submissions for External Infusion Pumps," issued March, 1993.

CITR Control Number: 0593-0026
Expiration Date: December 2, 2014

For questions regarding this document, please contact: Device Branch, Office of Device Evaluation at 301-988-6420 or visit www.cdrh.gov.
For questions regarding safety assurance cases, Device Branch, Office of Device Evaluation at 301-443-7547, or visit www.cdrh.gov.
For questions regarding pre-market clearance clinical studies, Food and Drug Administration, Center for Devices and Radiological Health, Compliance at 301-594-7180 or visit www.cdrh.gov.
For questions pertaining to manufacturers report 301-796-6420 or visit www.cdrh.gov.

For questions pertaining to manufacturers report 301-796-6420 or visit www.cdrh.gov.

CDRH

Support for Safety Case Generation via Model Transformation

Cheng Liang (1), Wanxi Shen (2), Zhuo Xie (3), Beatrice Hay (4), Xian-Qing Li (2), Joseph Aiton (1), Richard Hambric (5)
(1) Department of Computer Science University of York, York, UK (2) State Key Laboratory of Intelligent Services Computing, School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China, (3) HZI, Hannover, Germany, (4) University of York, York, UK, (5) Institute for Manufacturing, University of Cambridge, Cambridge, UK.

Abstract

S1: Functional safety is verified in "GPCA system"
G1: Operational safety is verified in "GPCA system"
S2: All related scenarios are given by "SRI1.2", "SRI1.3", "SRI1.4", and "SRI5.5"
S3: All operational hazards are mitigated
G2: All operational hazards are mitigated
S4: "Overrun" is mitigated
S5: "Underrun" is mitigated
S6: "Underrun" is mitigated under "Programmed flow rate too low"
S7: "Period is 15 min" definition is appropriate
S8: "Flow rate sensor is equipped", period is 15 min, flow rate less than 90% of the programmed rate
S9: "Flow rate sensor is equipped", period is 15 min, flow rate less than 90% of the programmed rate

Figure 9 Safety case model of GPCA system

Footnotes

1. Based on FDA studies of pump devices, FDA requires that the pump motor and feedback mechanism of pumps exceed target reliability in order to assure safe and effective devices. The minimum requirements include a minimum of 99% reliability, with a minimum of 95% confidence interval, and a minimum of 100,000 hours of performance. The criteria are determined based on the specific use case of the pump device.

2. The lift capacity of the pump mechanism should be selected based on the maximum flow rate to be delivered by the infusion pump, the patient's body weight, and the compliance of the infusion system.

3. The lift capacity of the pump mechanism should be selected based on the maximum flow rate to be delivered by the infusion pump, the patient's body weight, and the compliance of the infusion system.

4. The lift capacity of the pump mechanism should be selected based on the maximum flow rate to be delivered by the infusion pump, the patient's body weight, and the compliance of the infusion system.

5. The lift capacity of the pump mechanism should be selected based on the maximum flow rate to be delivered by the infusion pump, the patient's body weight, and the compliance of the infusion system.

6. The lift capacity of the pump mechanism should be selected based on the maximum flow rate to be delivered by the infusion pump, the patient's body weight, and the compliance of the infusion system.

7. The lift capacity of the pump mechanism should be selected based on the maximum flow rate to be delivered by the infusion pump, the patient's body weight, and the compliance of the infusion system.

8. The lift capacity of the pump mechanism should be selected based on the maximum flow rate to be delivered by the infusion pump, the patient's body weight, and the compliance of the infusion system.
The Assurance Case

Medical
Space
Aeronautics
Rail
Automotive
Shipping
Autonomous
Critical Infrastructure
Cyber Physical Systems...

Dependability
Engineering
Innovation for Cyber Physical Systems
Communicating Assurance to Gain Trust

ISO/IEC 15026-2 Assurance Case
OMG Structured Assurance Case Metamodel (SACM)
Figure 1. Top Level of an Assurance Case

- Confidentiality
- Integrity
- Availability
- Access control
- Assets & threat actors identified & addressed
- Identification
- Authentication
- Authorization

System is adequately secure against moderate threats

Security requirements identified and met by functionality

Security implemented by software life cycle processes

See figure 2
Apportionment of Ownership:
White = Generic, Green = Air Command, Orange = DE&S, Red = Contractors
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
TRANSPARENT ASSURANCE AS A BASIS FOR TRUST - FUTURE
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”

Questions?