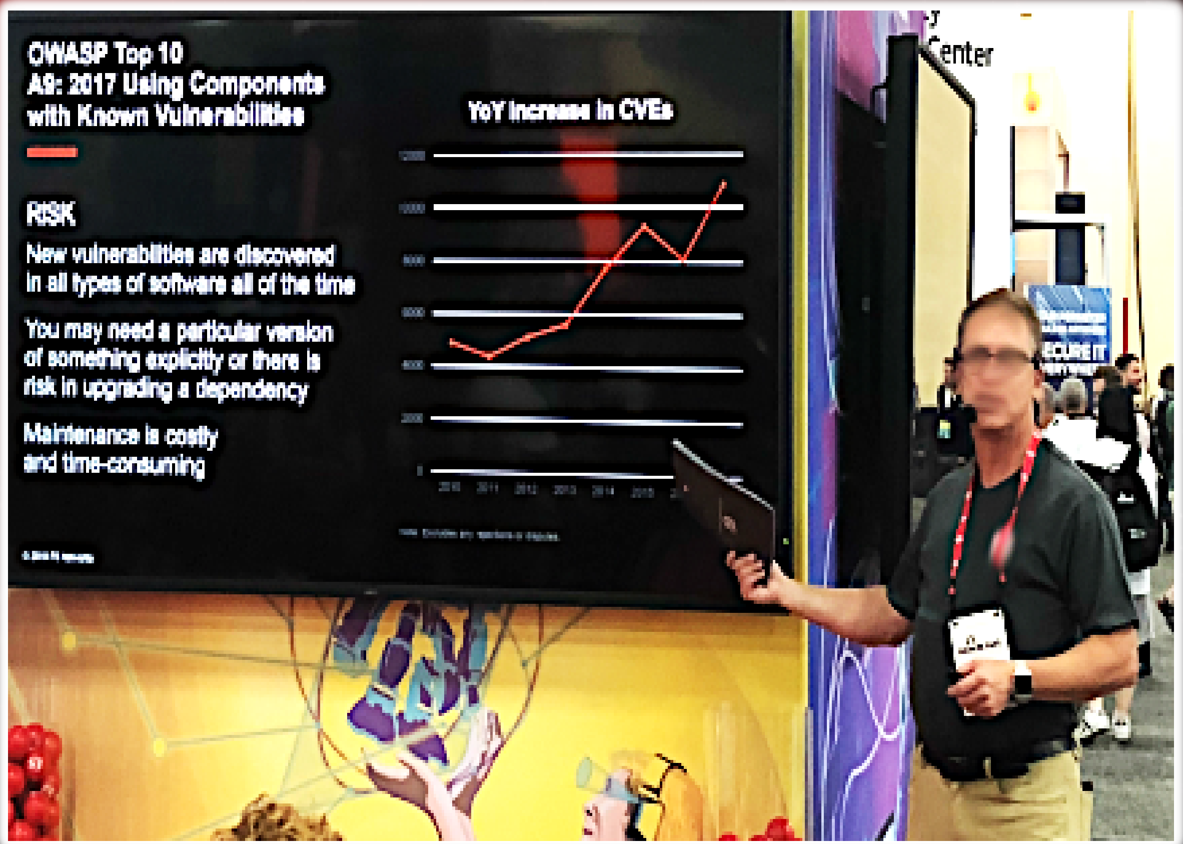
Visibility & Control: Addressing Supply Chain Challenges to Trustworthy Software-Enabled Things

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


OWASP Top 10
A9: 2017 Using Components with Known Vulnerabilities

RISK

New vulnerabilities are discovered in all types of software all of the time

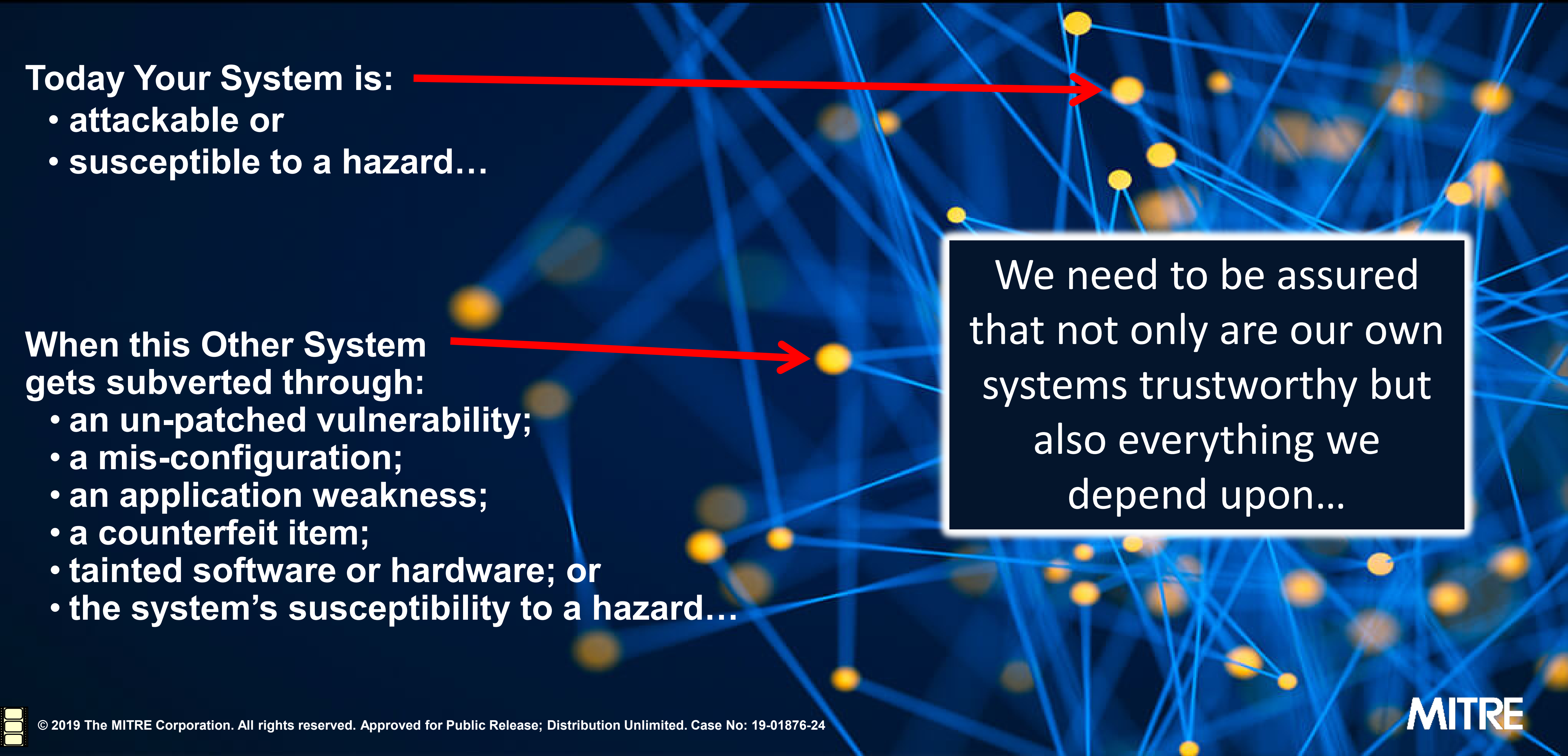
You may need a particular version of something explicitly or there is risk in upgrading a dependency

Maintenance is costly and time-consuming

IIC Forum | Long Beach, CA

MITRE

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…

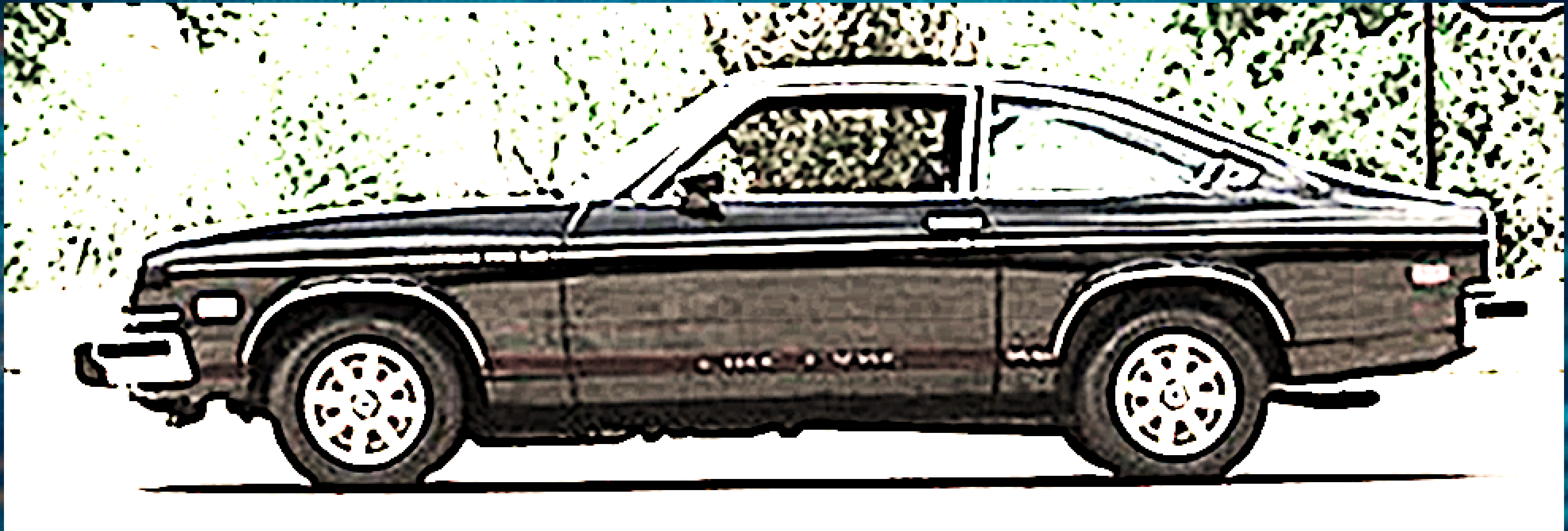
When this Other System gets subverted through:

- 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…

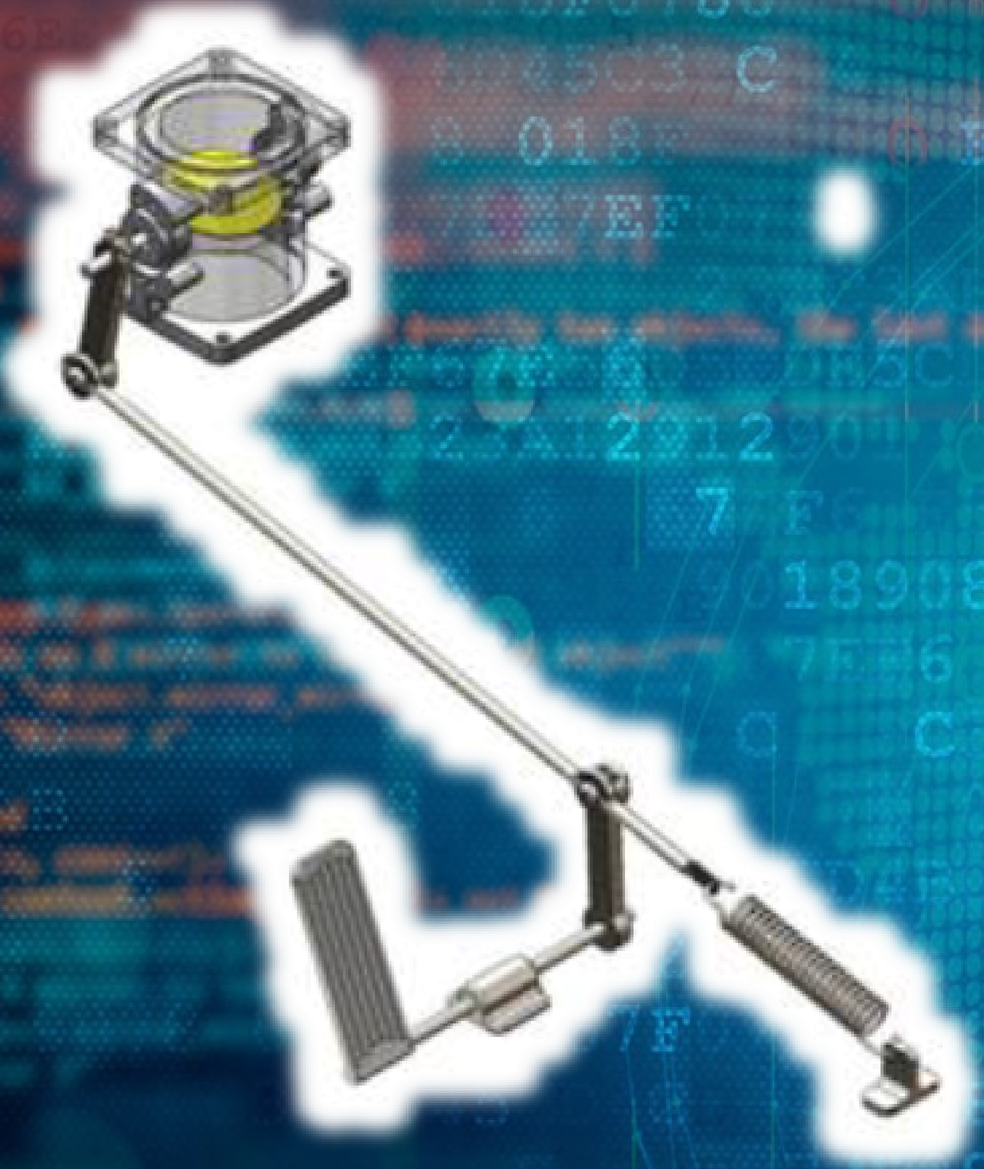
How is Software-Enabled and Connected (aka Cyber) Becoming so Pervasive?

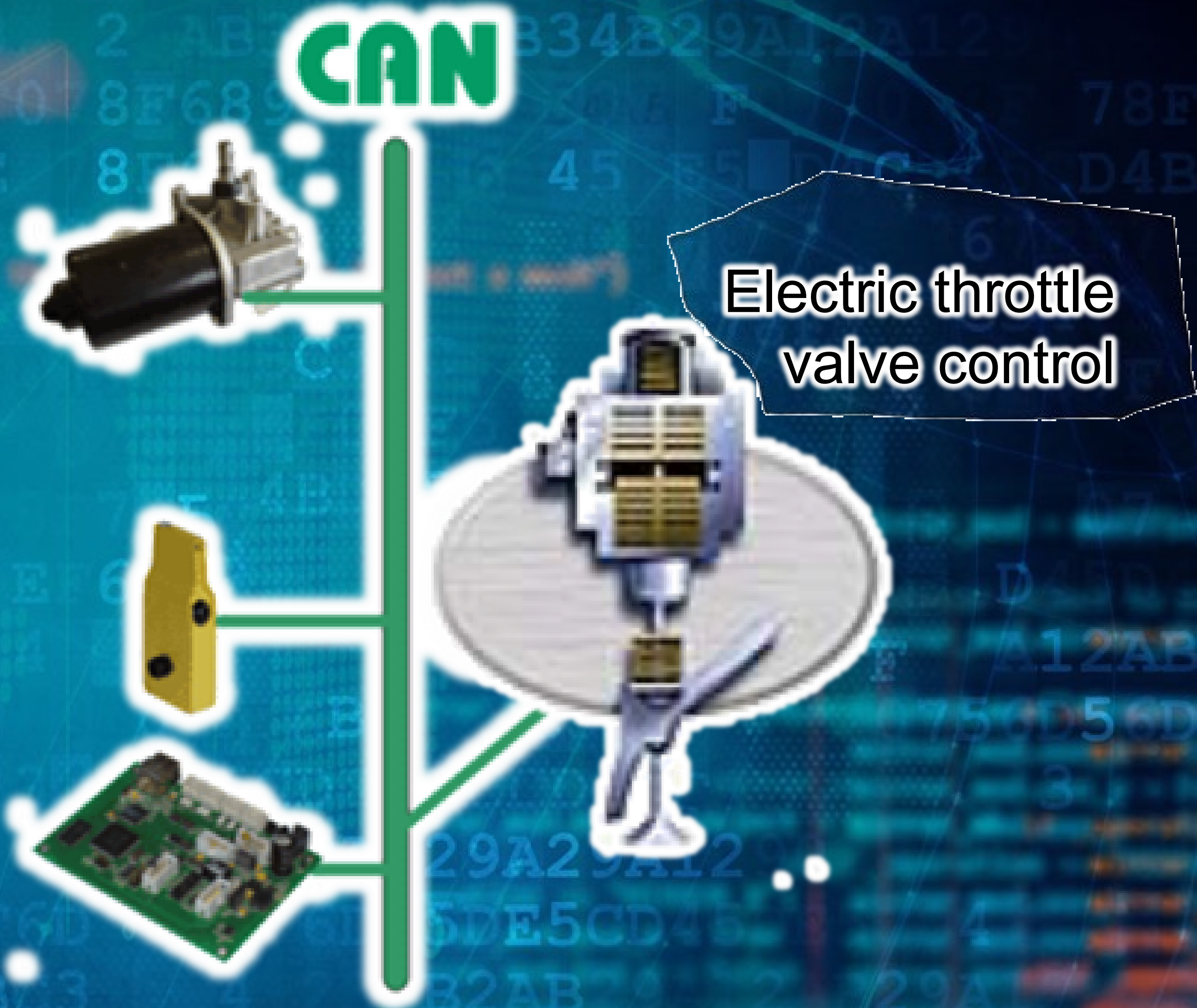
Context: 1976 Chevy Vega



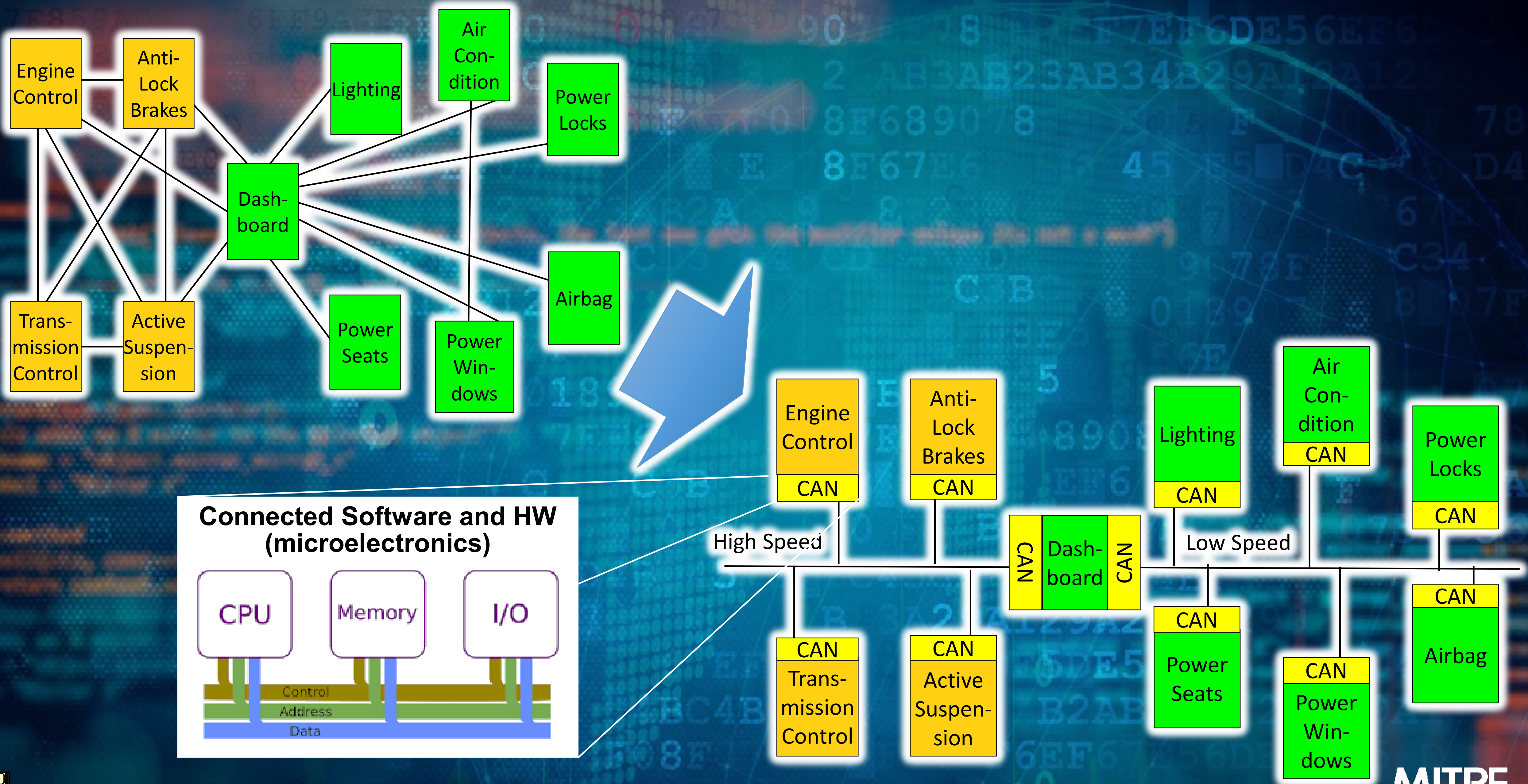
The only software was behind the wheel – no microelectronics.

Critical Functions Migrated into Software & Microelectronics (SW/HW)

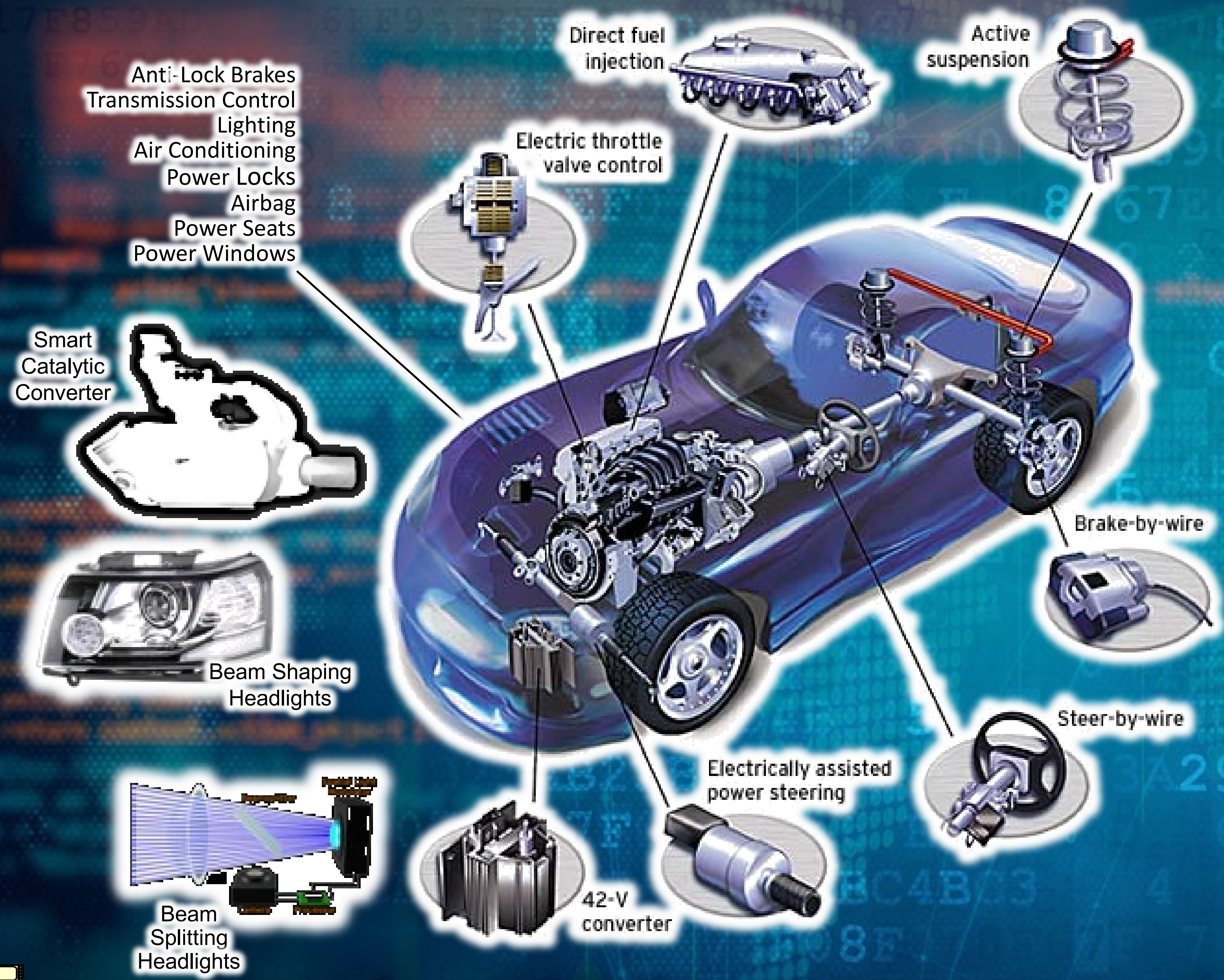


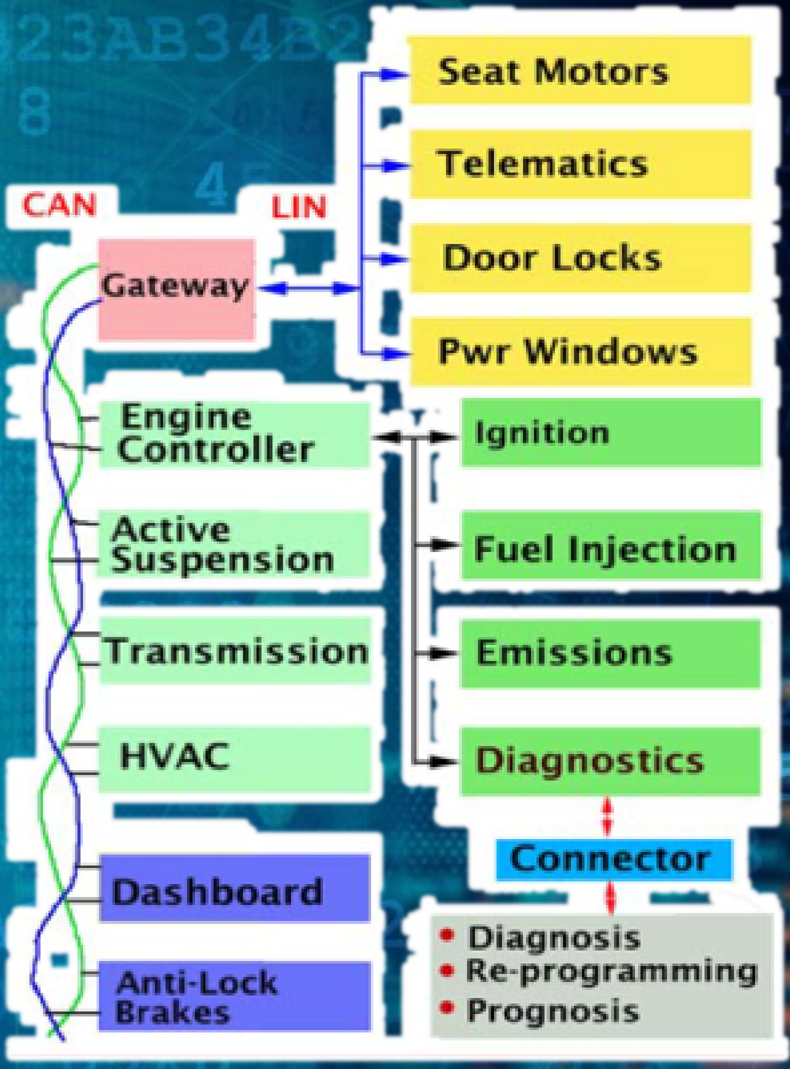


Control/Communication Transitioned from Point-to-Point Wiring to Network-based

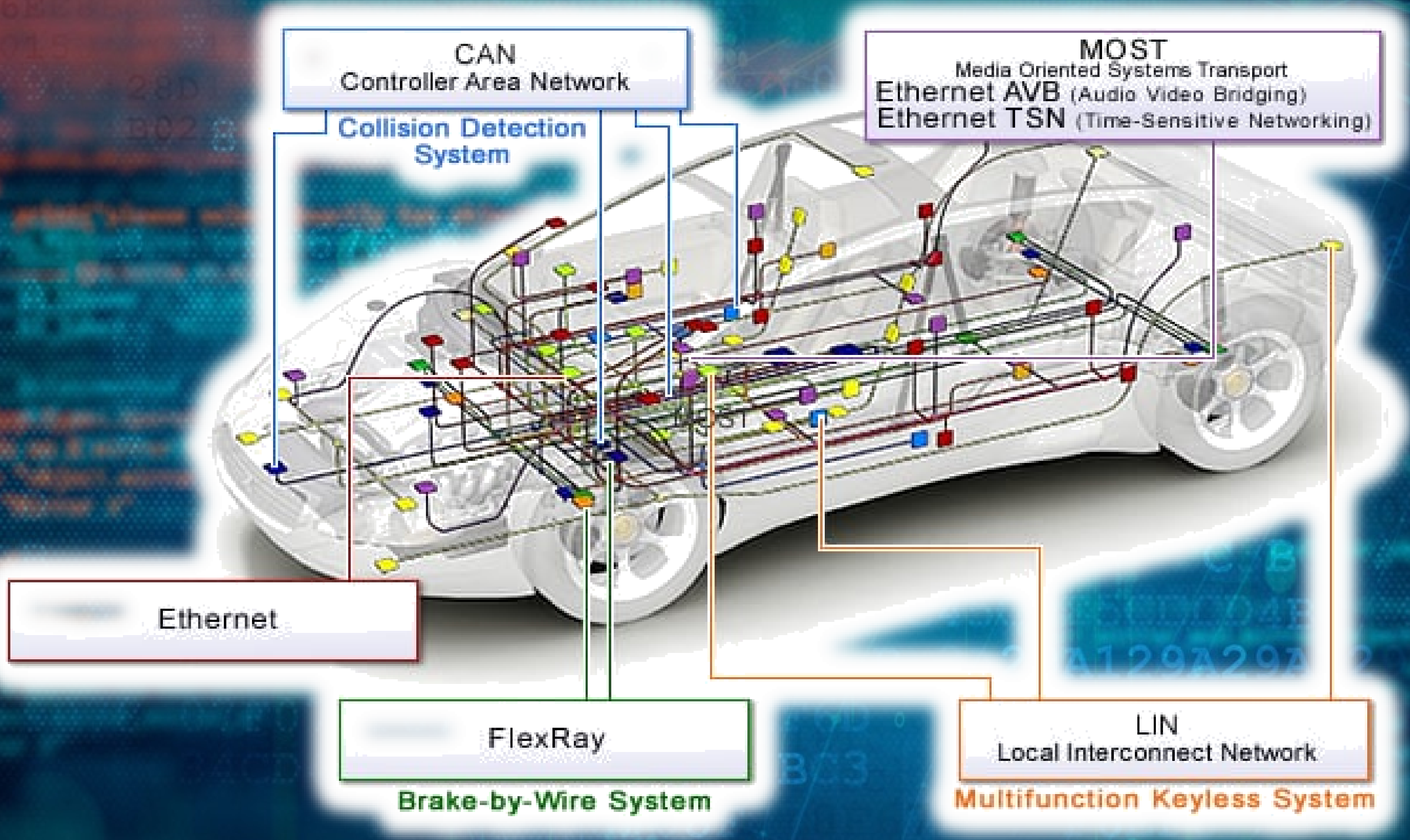


Critical Functions Are Migrating into Connected SW/HW

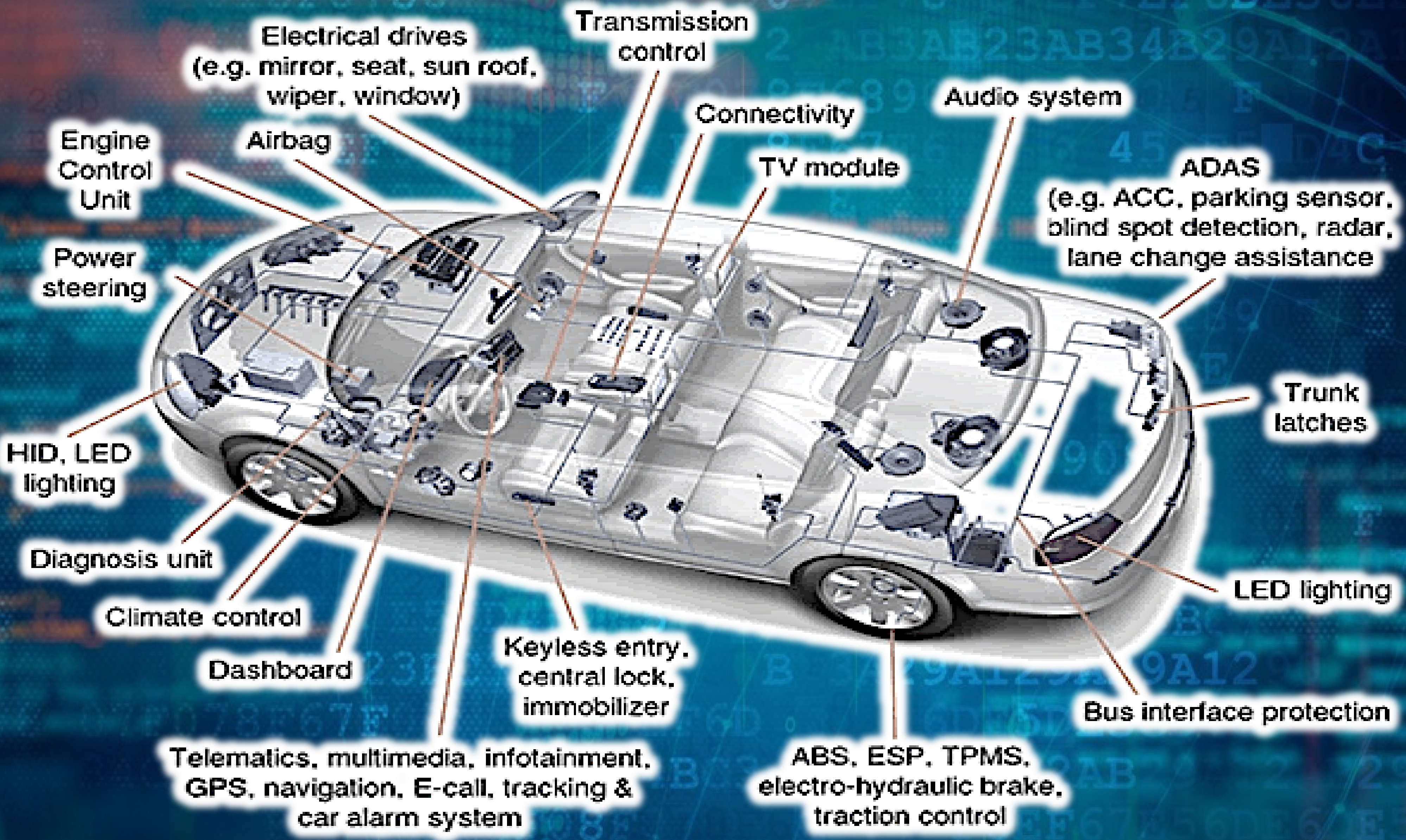




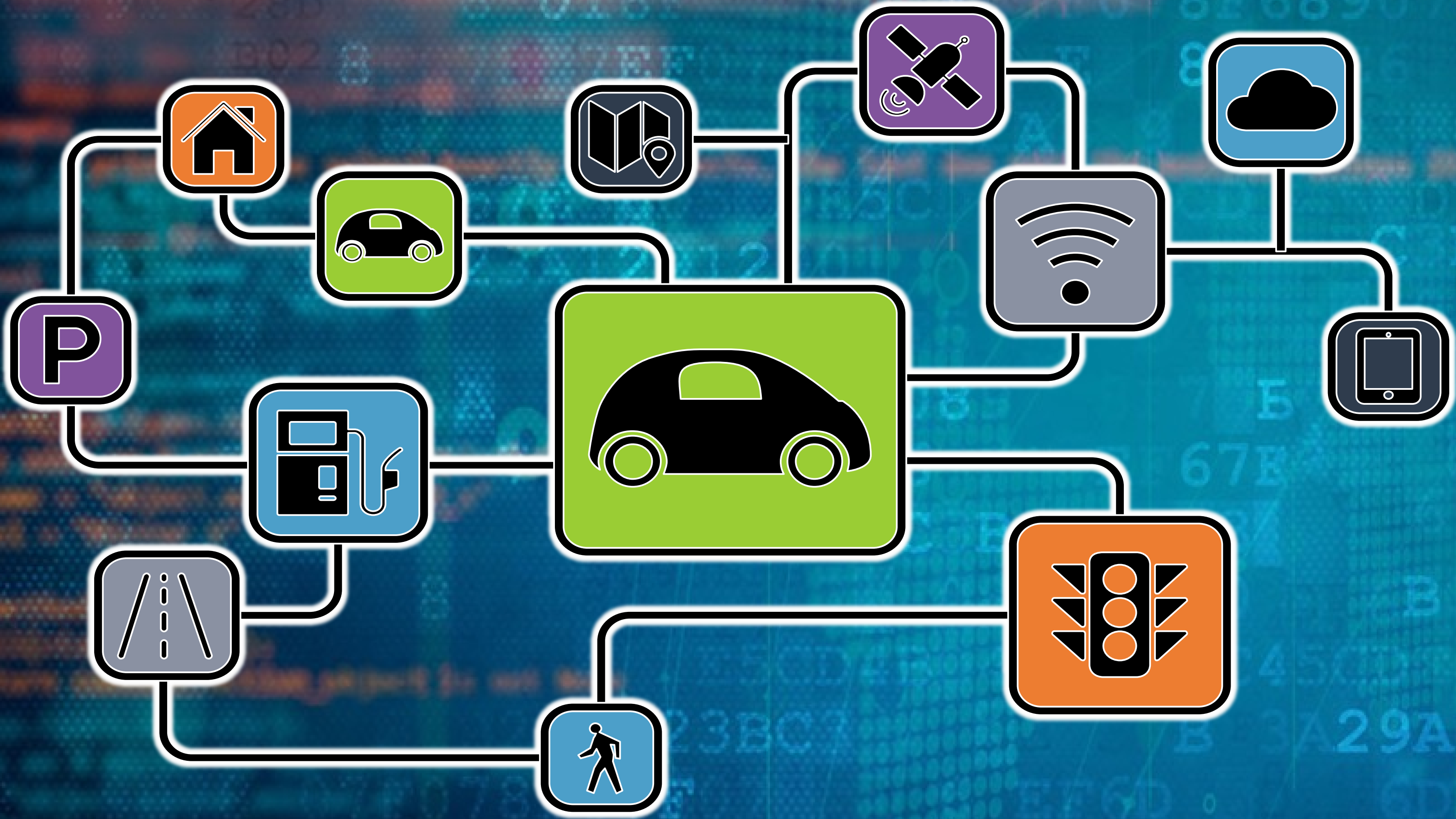
Multiple Types of Networks Are Appearing

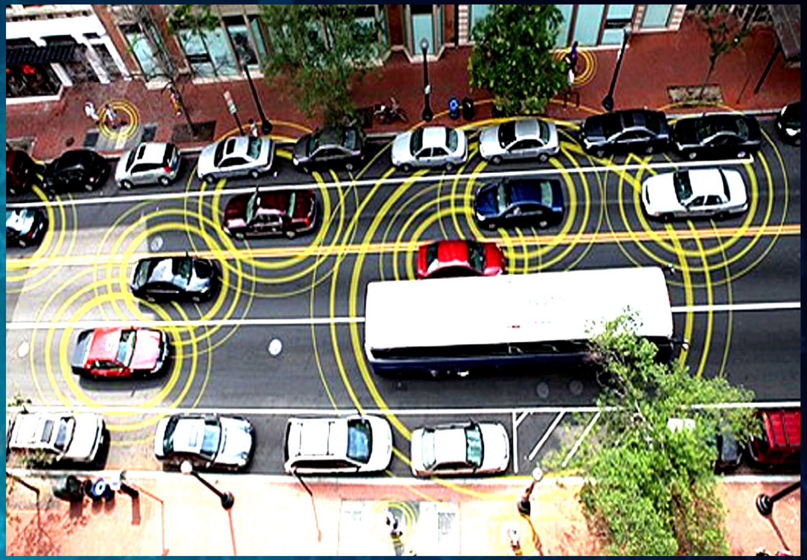

CAN
Controller Area Network
Collision Detection System
MOST
Media Oriented Systems Transport
Ethernet AVB (Audio Video Bridging)
Ethernet TSN (Time-Sensitive Networking)
Ethernet
FlexRay
Brake-by-Wire System
LIN
Local Interconnect Network
Multifunction Keyless System

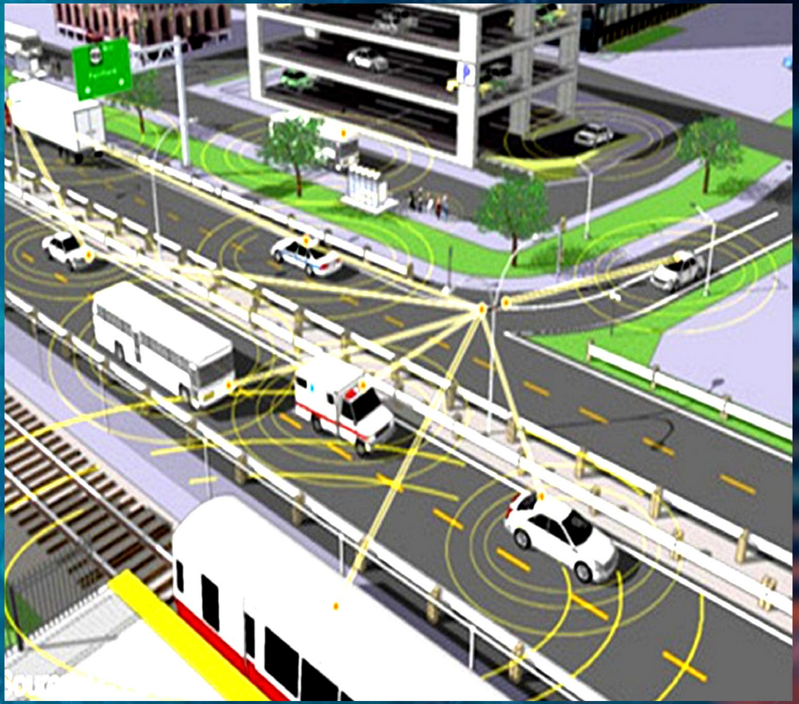
Many Critical Functions Now Need to be Updated and Sustained…


Electrical drives
(e.g. mirror, seat, sun roof, wiper, window)
Transmission control
Engine Control Unit
Airbag
Connectivity
Audio system
TV module
ADAS
(e.g. ACC, parking sensor, blind spot detection, radar, lane change assistance
Power steering
Trunk latches
HID, LED lighting
Diagnosis unit
LED lighting
Climate control
Dashboard
Keyless entry, central lock, immobilizer
Bus interface protection
Telematics, multimedia, infotainment, GPS, navigation, E-call, tracking & car alarm system
ABS, ESP, TPMS, electro-hydraulic brake, traction control

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

Vehicles

Buildings

Temperature, Humidity,CO2

Motion Sensor

AC,Chiller

Electric power

Elevator

Entrance gate

Aeronautics

Energy

Manufacturing

Shipping

These Changes Go Well beyond Traditional Information Technology…

Water Treatment

Status & Health Monitoring

Smart Munitions

Remote Management

Oil & Gas

Hydro Power & Dam Mngt

Secure Behavior

Reliable Behavior

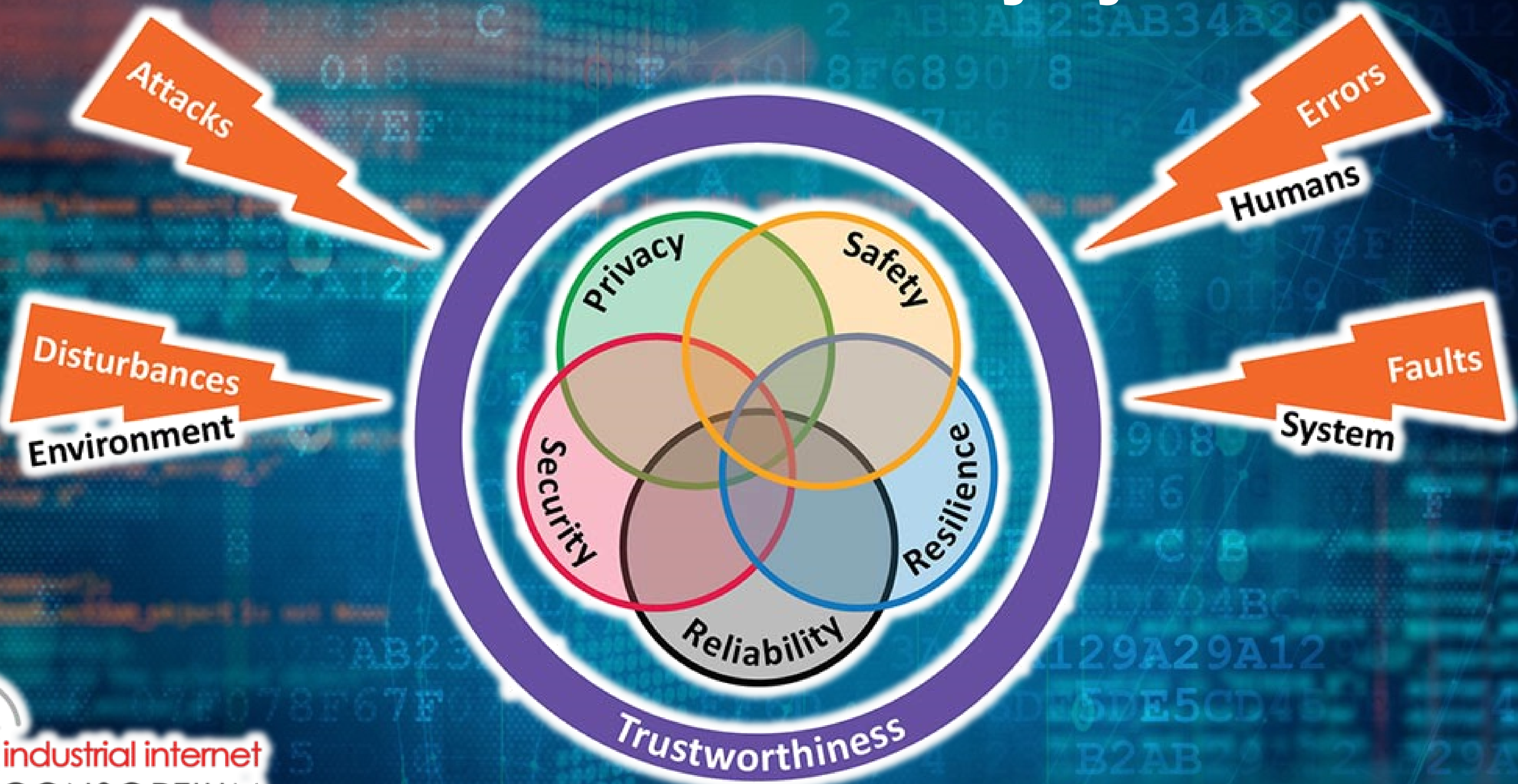
MIND THE GAP

Safe Behavior

Resilient Behavior

Privacy Expectations

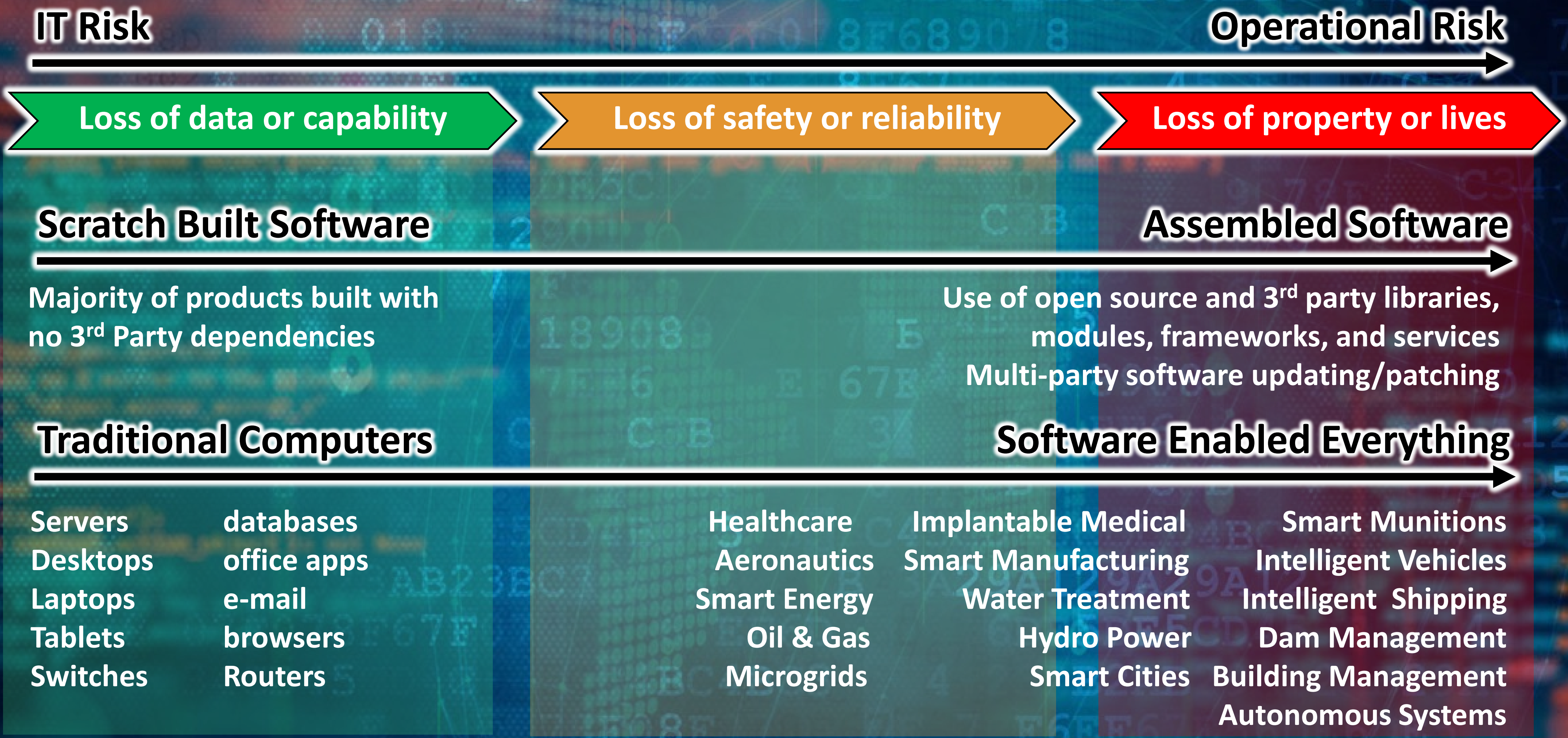
– Need Assurance of More Than Security –
Need Assured Trustworthy Systems



industrial internet CONSORTIUM

MITRE

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



IT Risk → Operational Risk

Loss of data or capability

Loss of safety or reliability

Loss of property or lives

Scratch Built Software → Assembled Software

Majority of products built with no 3rd Party dependencies

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

Multi-party software updating/patching

Traditional Computers → Software Enabled Everything

Servers
Desktops
Laptops
Tablets
Switches

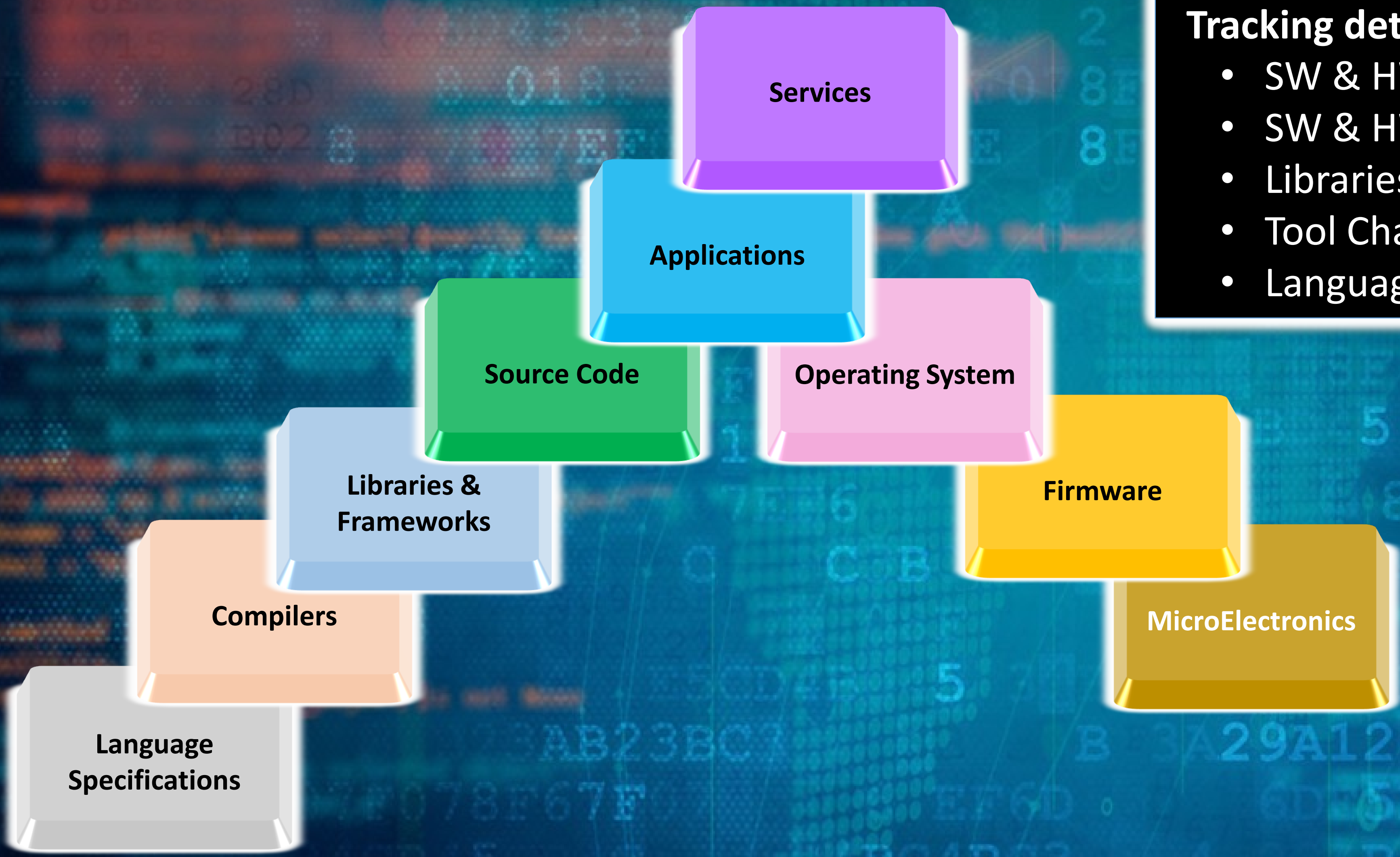
databases
office apps
e-mail
browsers
Routers

Healthcare
Aeronautics
Smart Energy
Oil & Gas
Microgrids

Implantable Medical
Smart Manufacturing
Water Treatment
Hydro Power
Smart Cities

Smart Munitions
Intelligent Vehicles
Intelligent Shipping
Dam Management
Building Management
Autonomous Systems

For Software-Enabled IIoT Version Control is Crucial

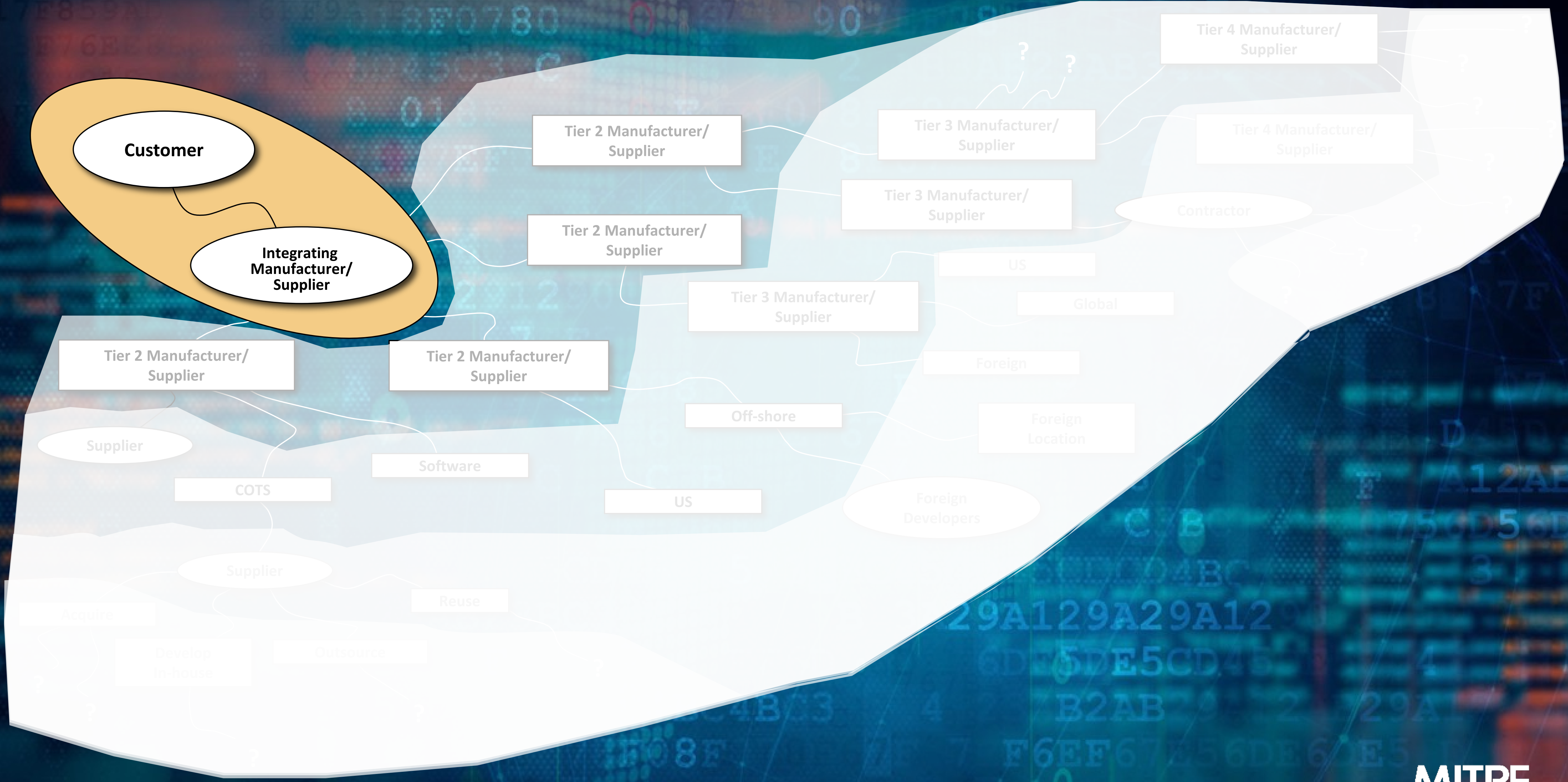


Tracking details for SW & HW components

- SW & HW Part numbers/names
- SW & HW versions
- Libraries & Frameworks Used
- Tool Chain Used/Flags/Options
- Languages & versions used

Attacks
Errors
Humans
Disturbances
Environment
Faults
System
Privacy
Safety
Security
Resilience
Reliability
Trustworthiness

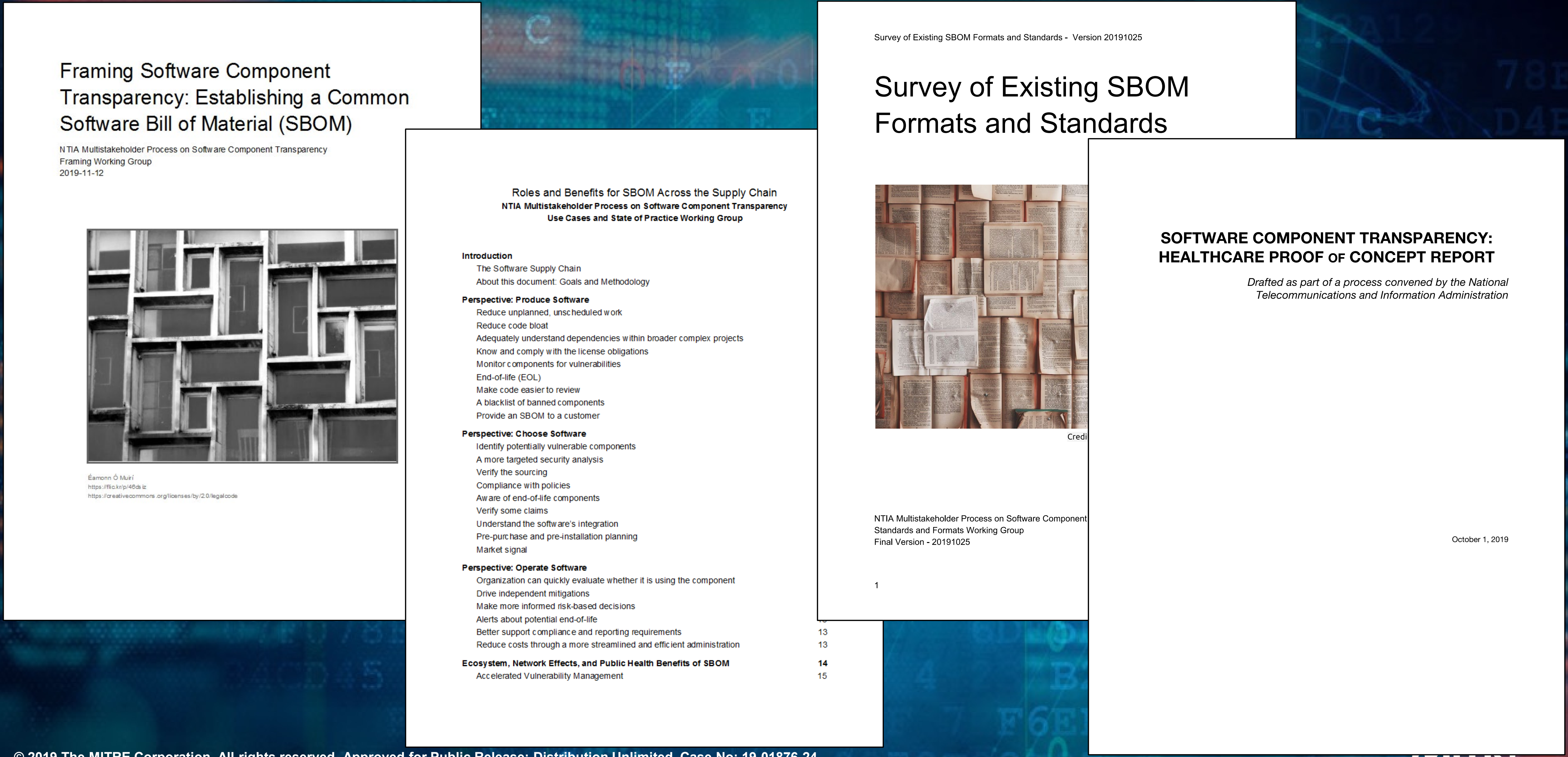
The Supply Chain for Software-Enabled Capabilities is Opaque


Customer
Integrating Manufacturer/ Supplier
Tier 2 Manufacturer/ Supplier
Tier 2 Manufacturer/ Supplier
Tier 2 Manufacturer/ Supplier
Tier 2 Manufacturer/ Supplier
Tier 3 Manufacturer/ Supplier
Tier 3 Manufacturer/ Supplier
Tier 3 Manufacturer/ Supplier
Tier 4 Manufacturer/ Supplier
Tier 4 Manufacturer/ Supplier
?
?
Contractor
US
Global
Foreign
Off-shore
Foreign Location
Foreign Developers
Supplier
Software
COTS
US
Supplier
Reuse
Acquire
Develop In-house
Outsource
?

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



Framing Software Component Transparency: Establishing a Common Software Bill of Material (SBOM)

NTIA Multistakeholder Process on Software Component Transparency
Framing Working Group
2019-11-12

Éamonn Ó Muirí
https://flic.kr/p/46dsiz
https://creativecommons.org/licenses/by/2.0/legalcode

Roles and Benefits for SBOM Across the Supply Chain
NTIA Multistakeholder Process on Software Component Transparency
Use Cases and State of Practice Working Group

Survey of Existing SBOM Formats and Standards - Version 20191025

Survey of Existing SBOM Formats and Standards

Credi

NTIA Multistakeholder Process on Software Component
Standards and Formats Working Group
Final Version - 20191025

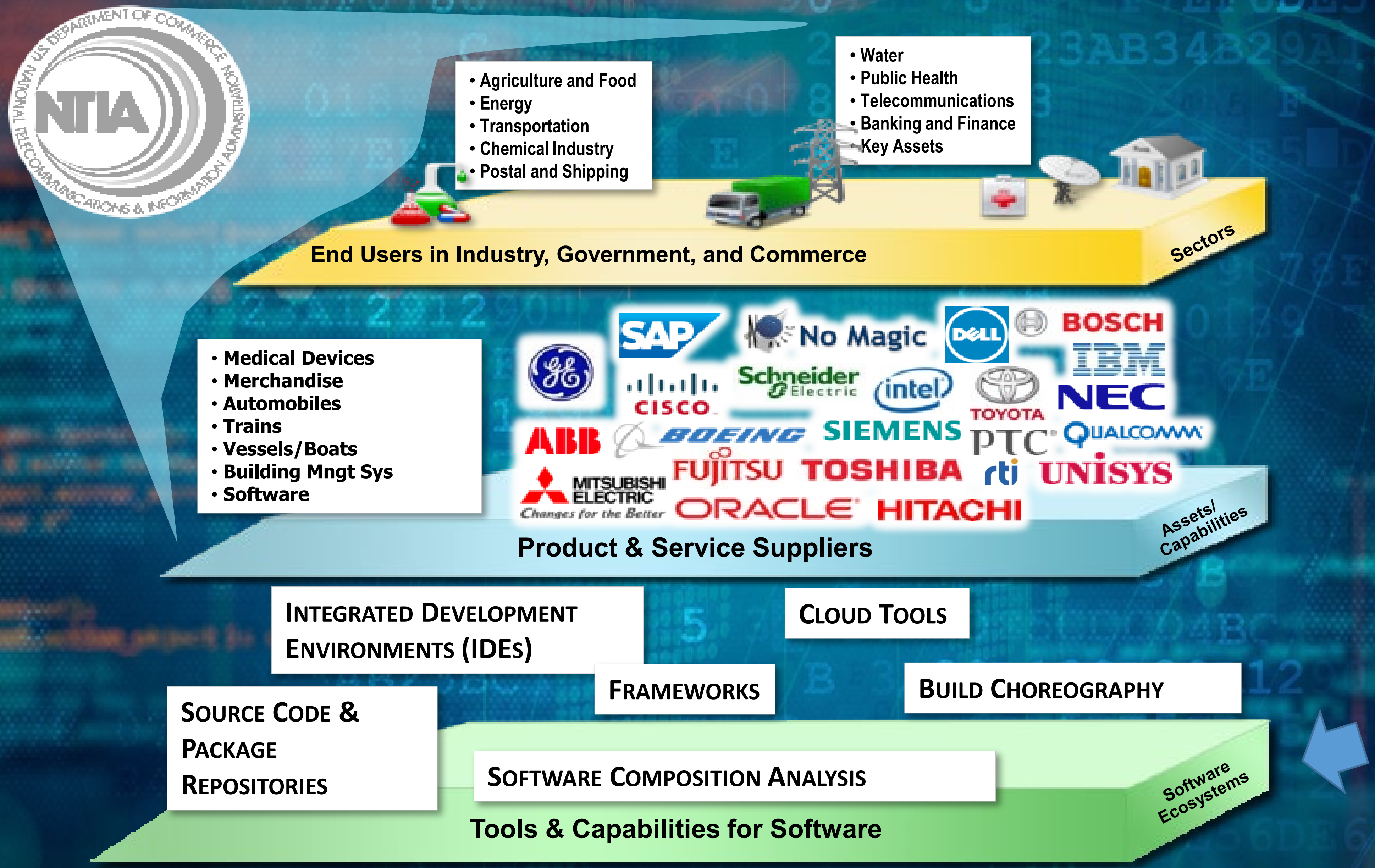
1

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

Lowering Adoption Hurdles for SBOMs



- Agriculture and Food
- Energy
- Transportation
- Chemical Industry
- Postal and Shipping

- Water
- Public Health
- Telecommunications
- Banking and Finance
- Key Assets

End Users in Industry, Government, and Commerce

Sectors

- Medical Devices
- Merchandise
- Automobiles
- Trains
- Vessels/Boats
- Building Mngt Sys
- Software

Product & Service Suppliers

Assets/ Capabilities

Integrated Development Environments (IDEs)

Cloud Tools

Frameworks

Build Choreography

Source Code & Package Repositories

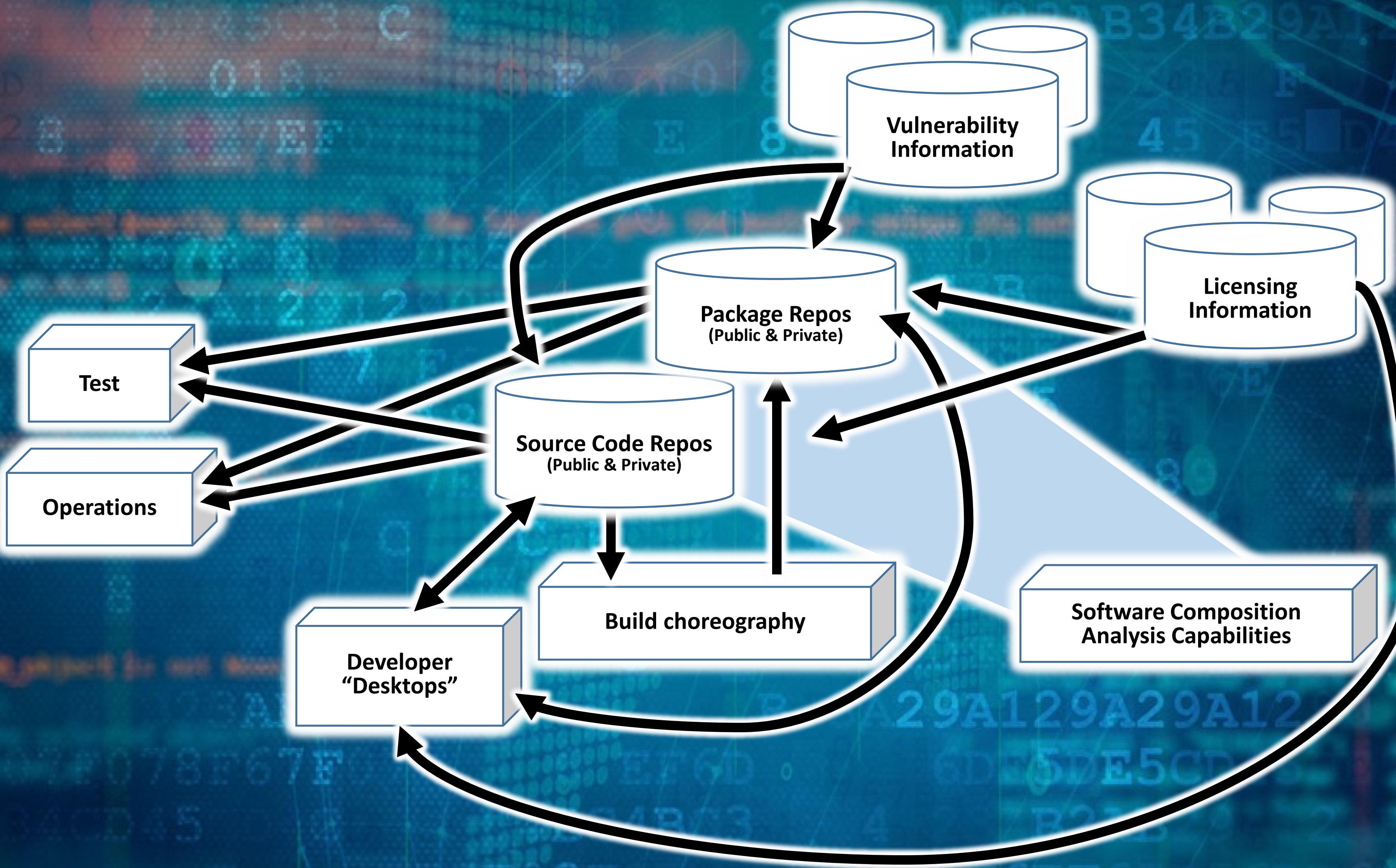
Software Composition Analysis

Tools & Capabilities for Software

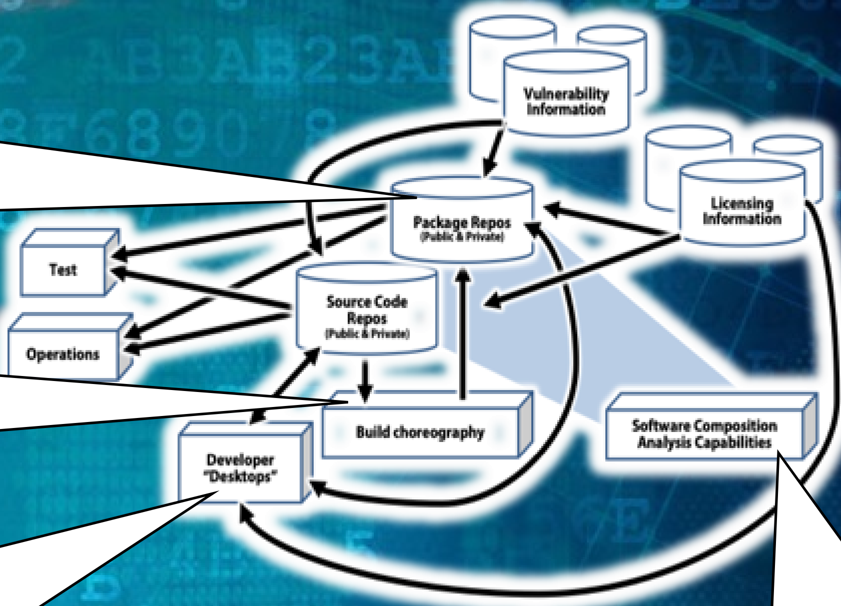
Software Ecosystems

Tool-to-Tool SBOM Exchange Standard effort

Ecosystem of SW Development, Integration, and Management Tools


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

SW Development, Integration, and Management Tools


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

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, CMake, 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)

IDEs: Android Studio, AppCode, Atom, BlueJ, CLion, Cloud9 IDE, Code Blocks, CodeCharge Studio, CodeLobster, CodePen, DataGrip, Eclipse, GoLand, IDLE, IntelliJ IDEA, LINX, Microsoft Visual Studio, MPLAB, NetBeans, PhpStorm, Pycharm, Rider, RubyMine, Spiralogics Application Architecture, WebStorm, Xcode, and Zend Studio

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)

Proper and Legal
(conditions about its use)

Pedigree
(history of how it was produced)

Known Sw Vulns
(known fixes are applied to it)

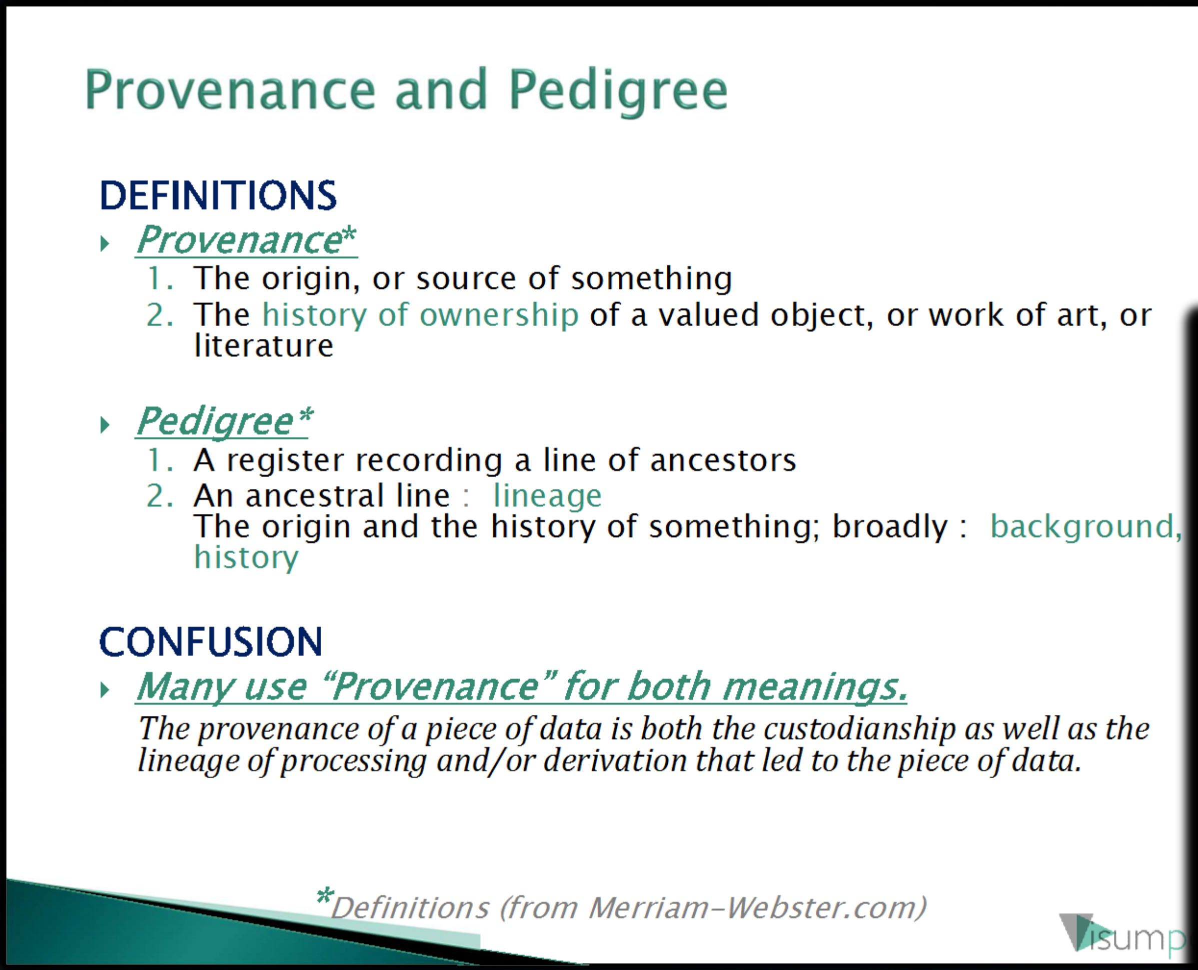
Provenance
(chain of custody of it)

Assurance
(safe-secure-resilient)

Integrity
(cryptographic basis of unalteredness)

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

Supply Chain Sequence Integrity



Provenance and Pedigree

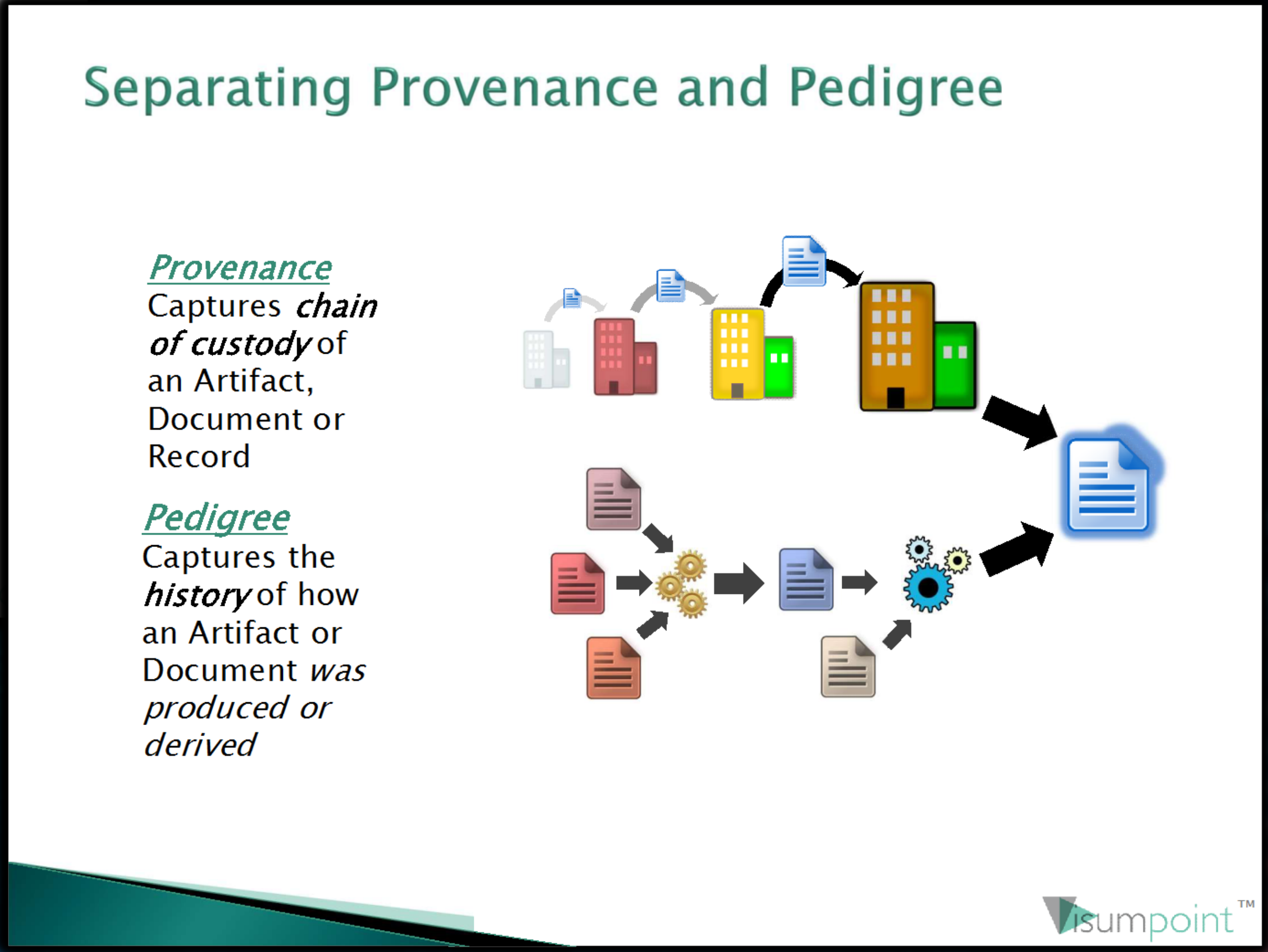
DEFINITIONS

- *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.

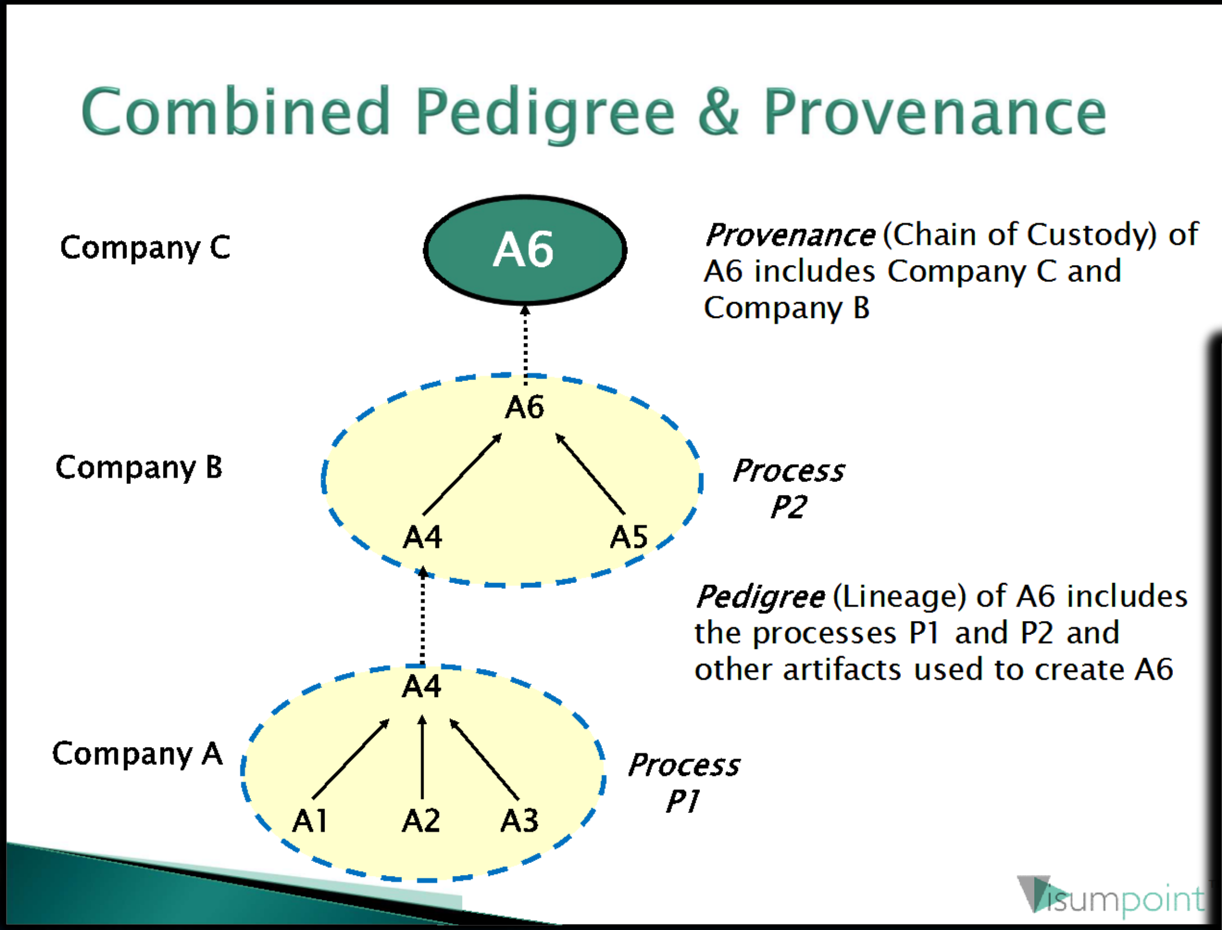
*Definitions (from Merriam-Webster.com)



Separating Provenance and Pedigree

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

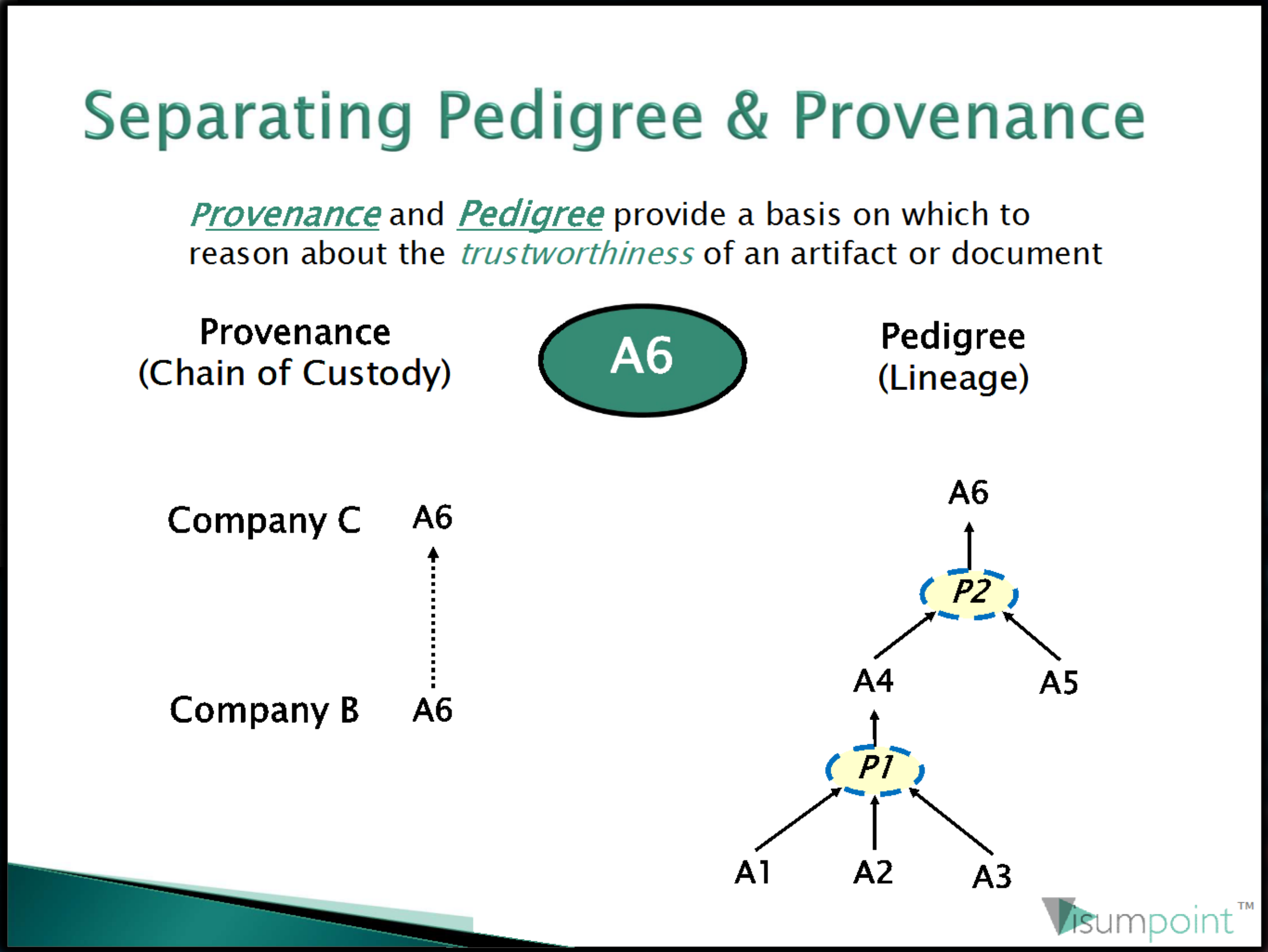
Pedigree
Captures the ***history*** of how an Artifact or Document *was produced or derived*



Combined Pedigree & Provenance

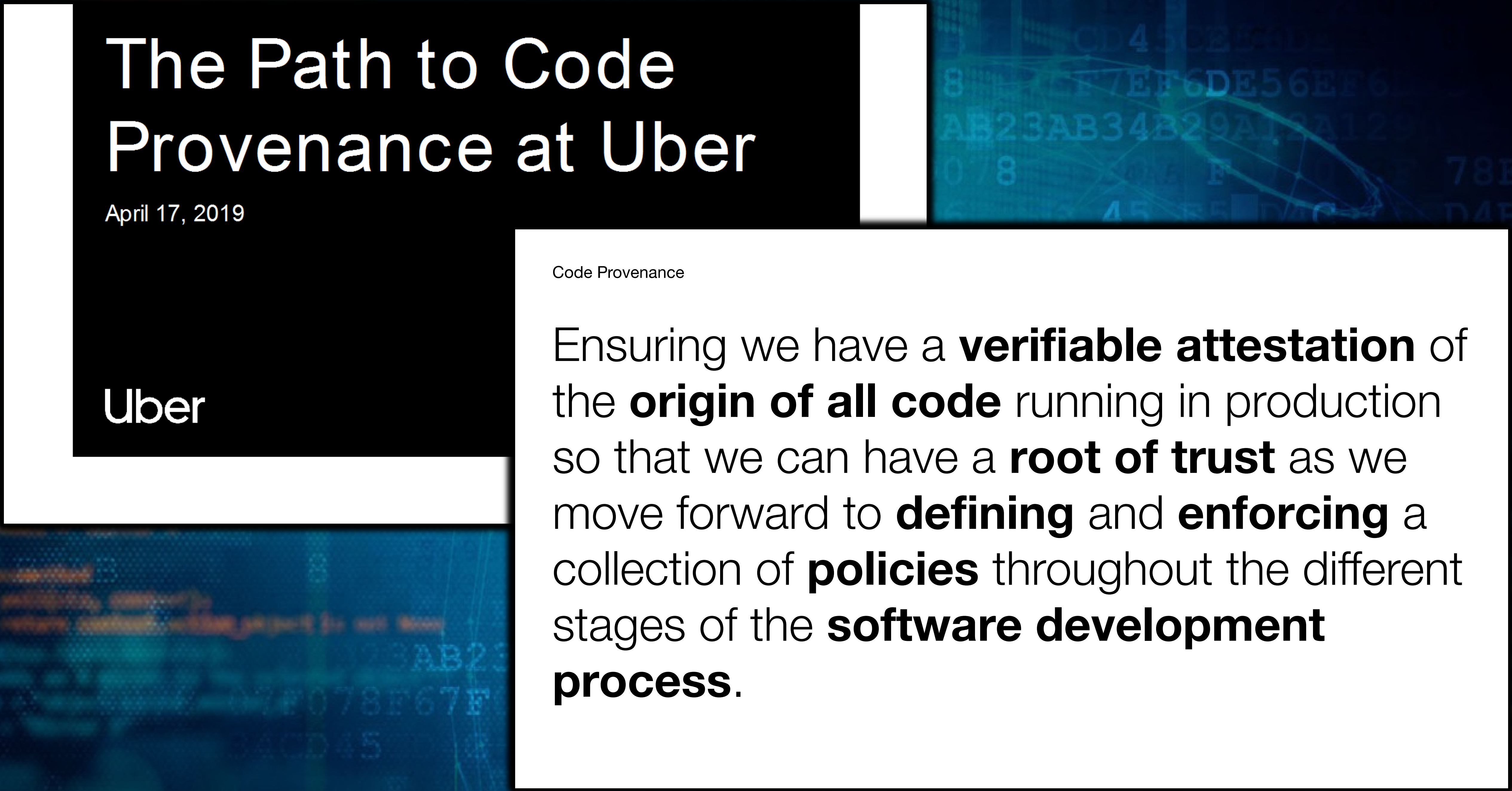
Provenance (Chain of Custody) of A6 includes Company C and Company B

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

Uber

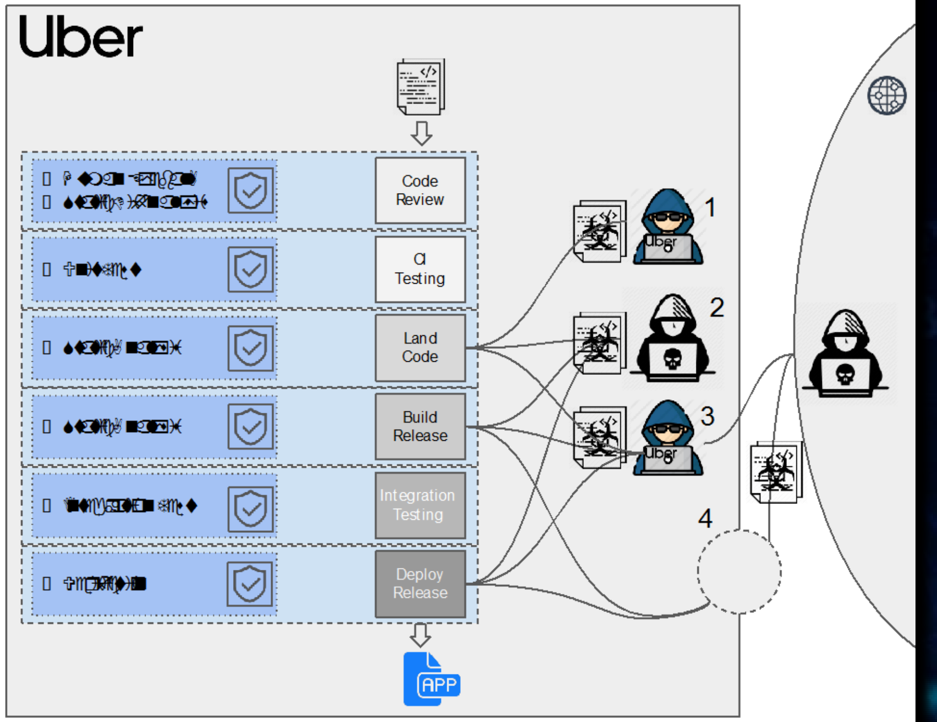
Code Provenance

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 are we protecting against?

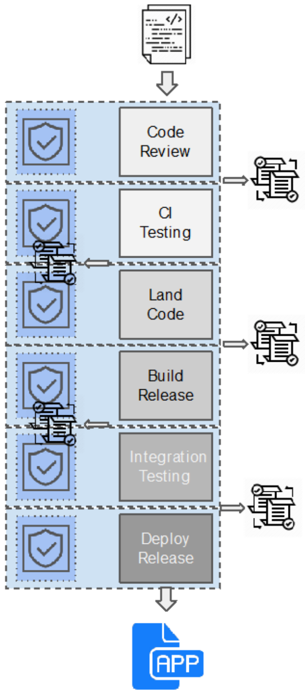
- [illegible]
- [illegible]
- [illegible]
- [illegible]



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



CISQ
Consortium for IT Software Quality

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

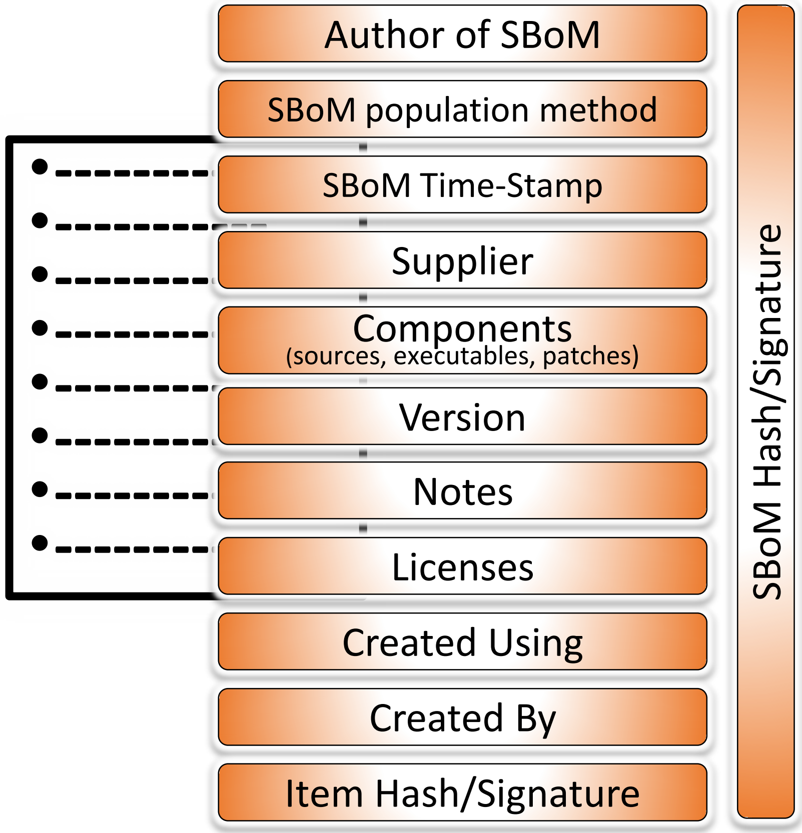
6 Known SW Vulns (known fixes are applied to it)

7 Assurance (secure-safe-resilient)

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

9 Supply Chain Sequence Integrity

SBoM elements

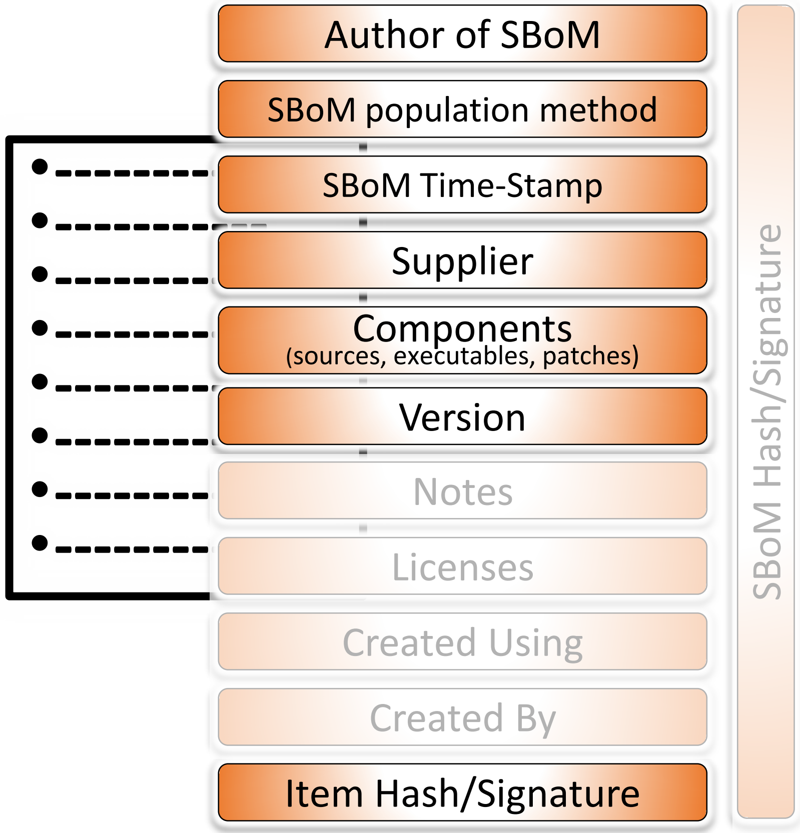


Correlated Info

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

SBoM Hash/Signature

Correlated Info

None

CISQ
Consortium for IT Software Quality

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

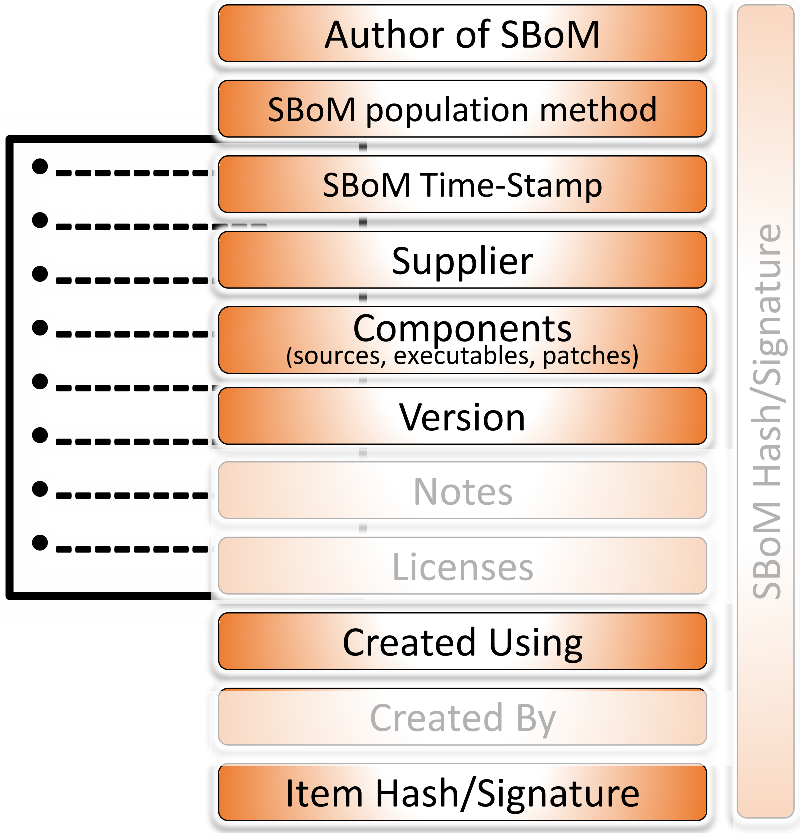
6 Known SW Vulns
(known fixes are applied to it)

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

SBoM 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

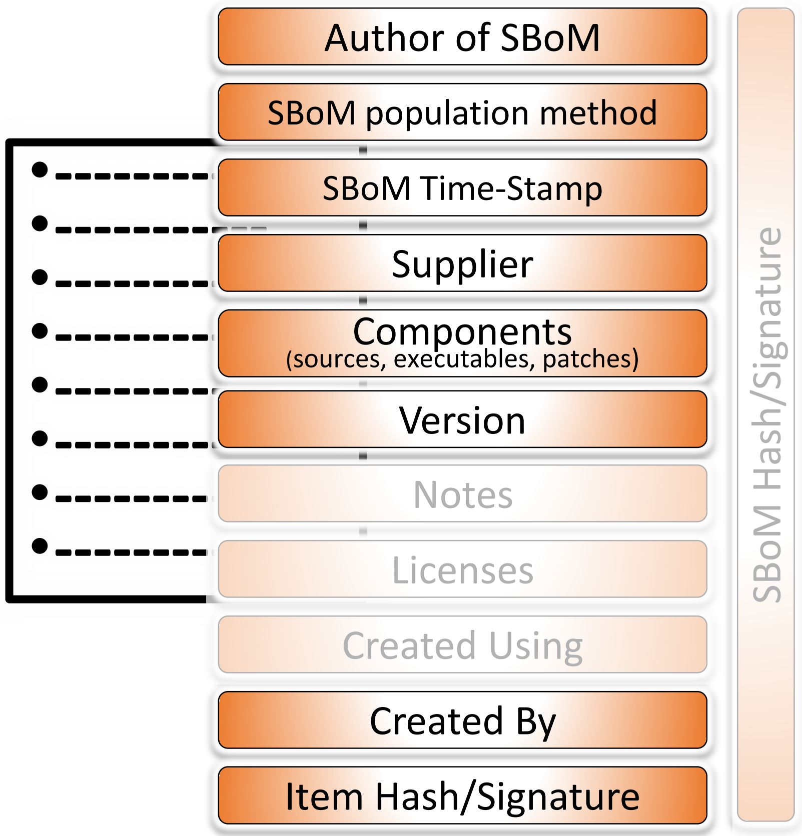
6 Known SW Vulns
(known fixes are applied to it)

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

SBoM 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

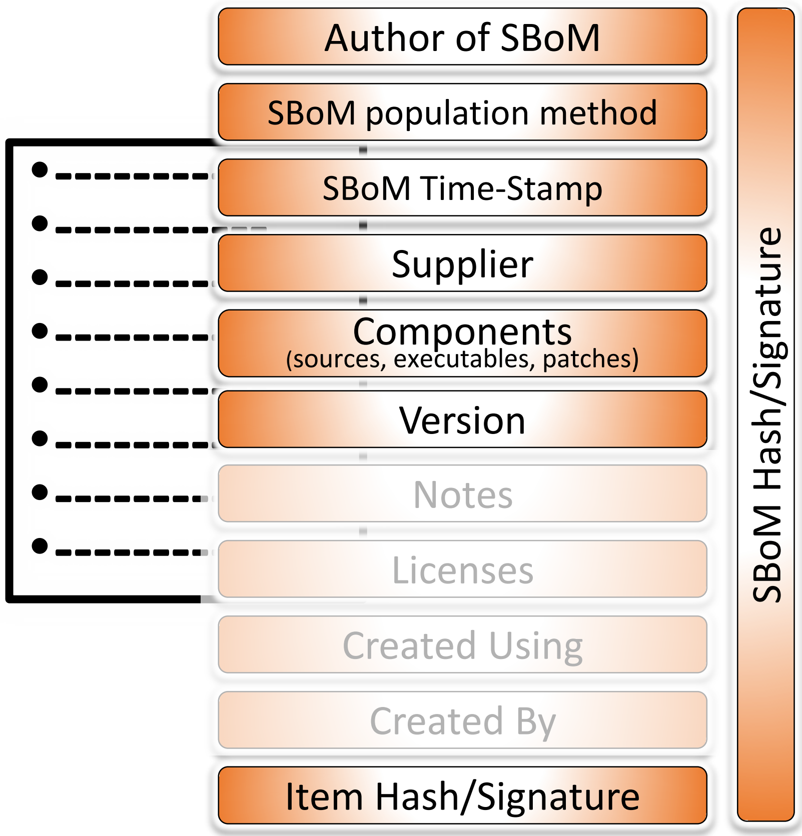
6 Known SW Vulns
(known fixes are applied to it)

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

SBoM 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

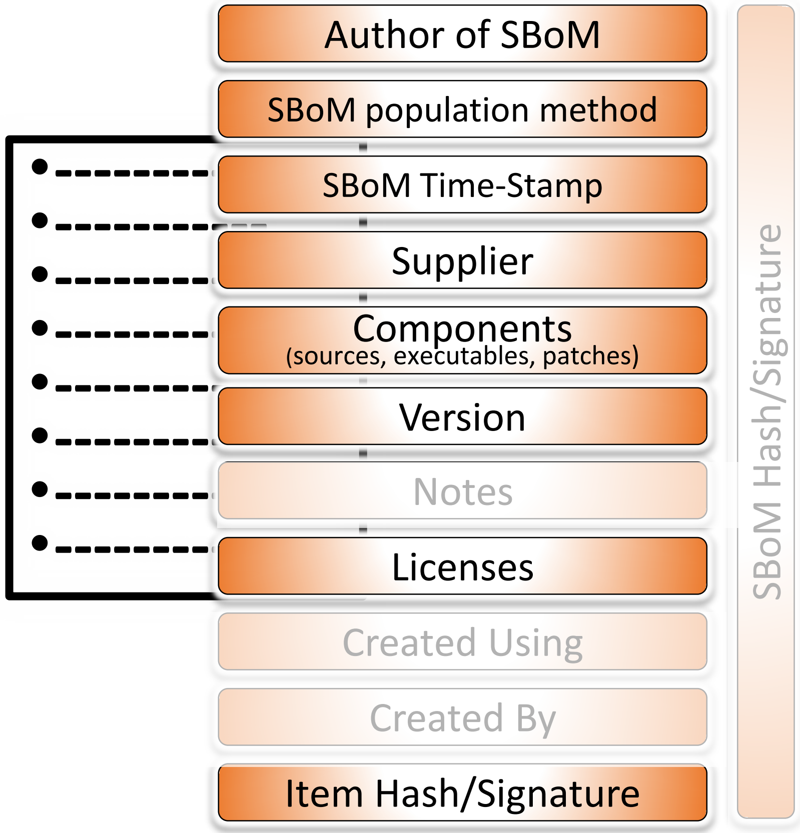
6 Known SW Vulns (known fixes are applied to it)

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

SBoM 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

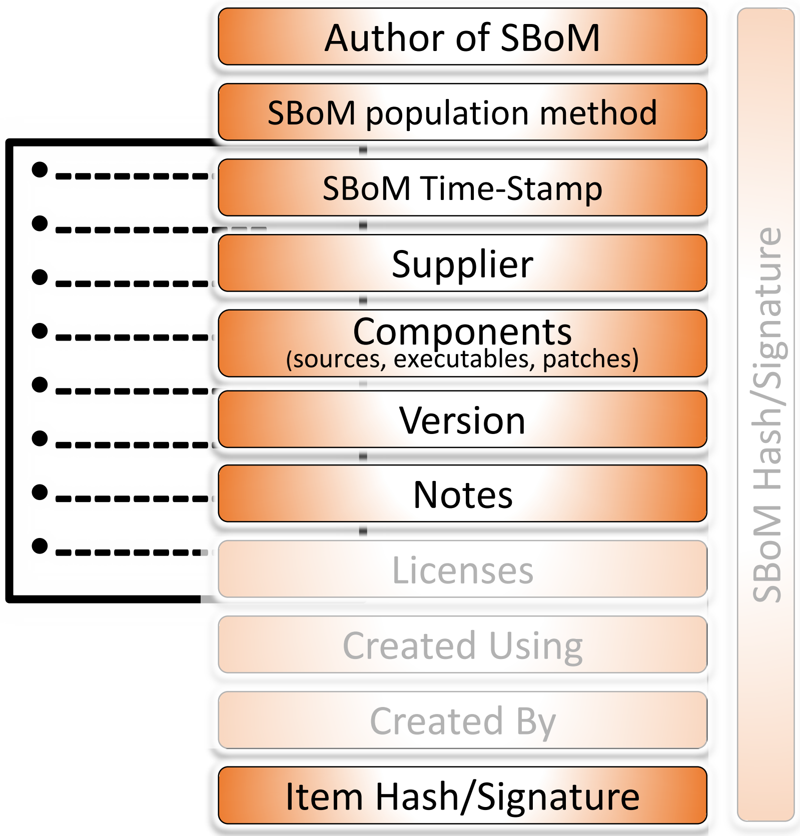
6 Known SW Vulns
(known fixes are applied to it)

7 Assurance
(secure-safe-resilient)

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

9 Supply Chain Sequence Integrity

SBoM elements



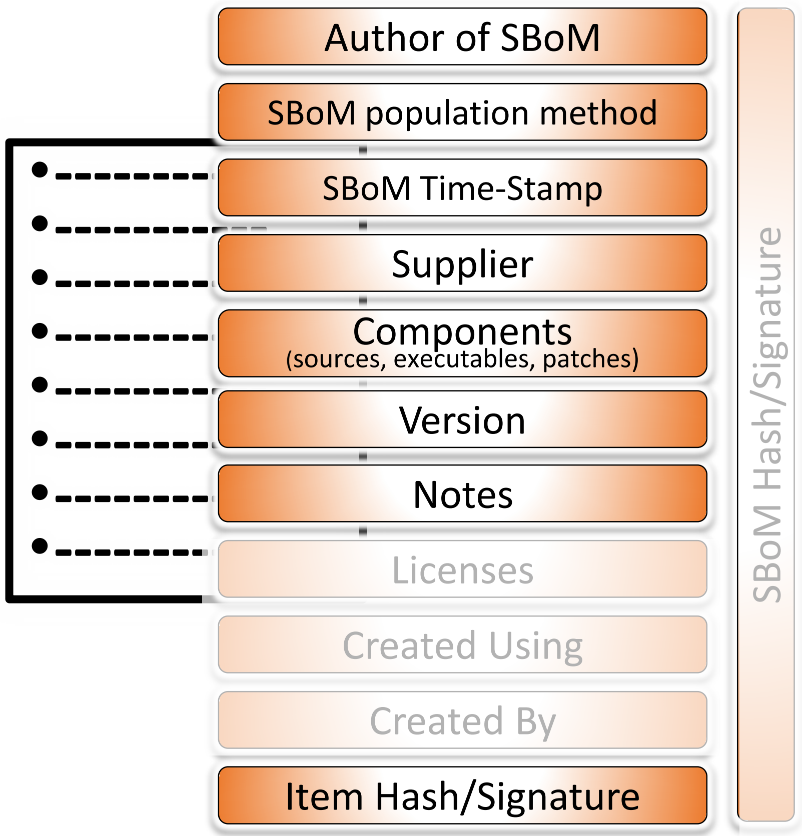
Correlated Info

Vulnerability Knowledge Bases
Vulnerability Management Systems

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

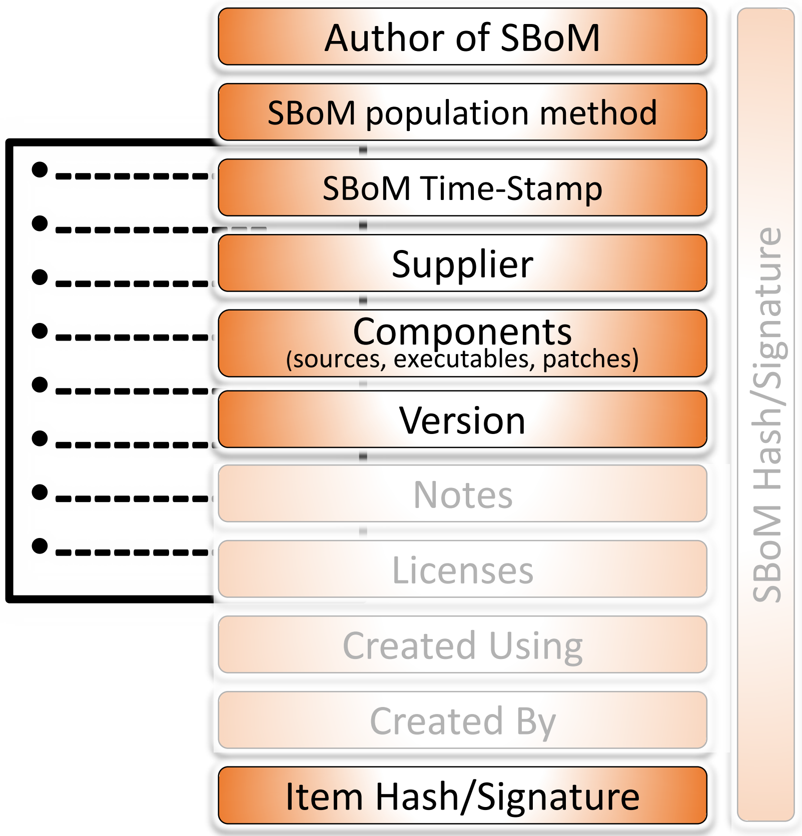
6 Known SW Vulns (known fixes are applied to it)

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

SBoM Hash/Signature

Correlated Info

Logging SBOMs of Services Used

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

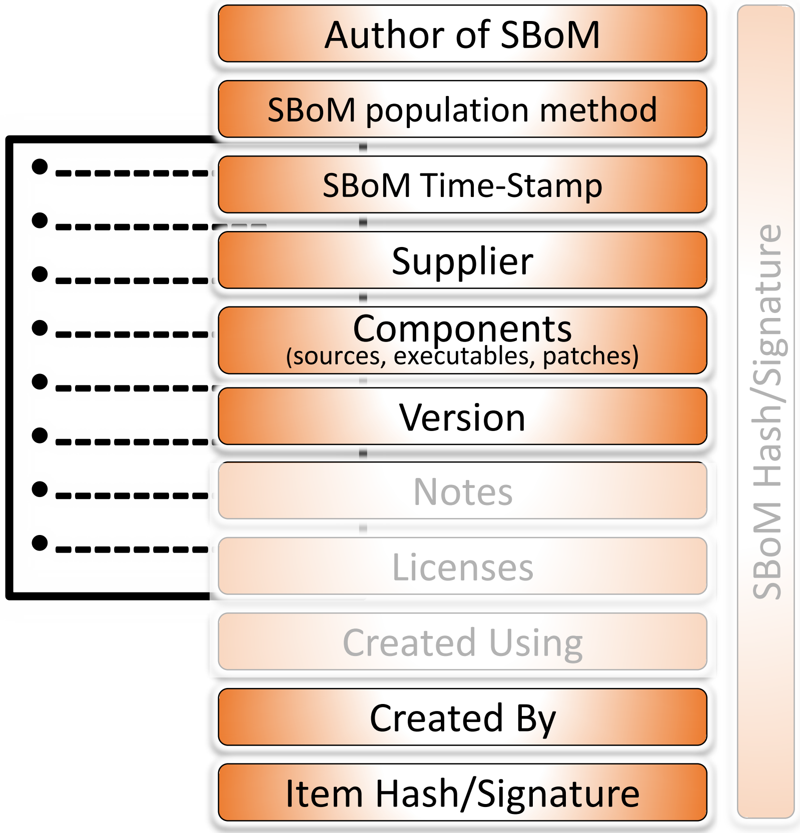
6 Known SW Vulns
(known fixes are applied to it)

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

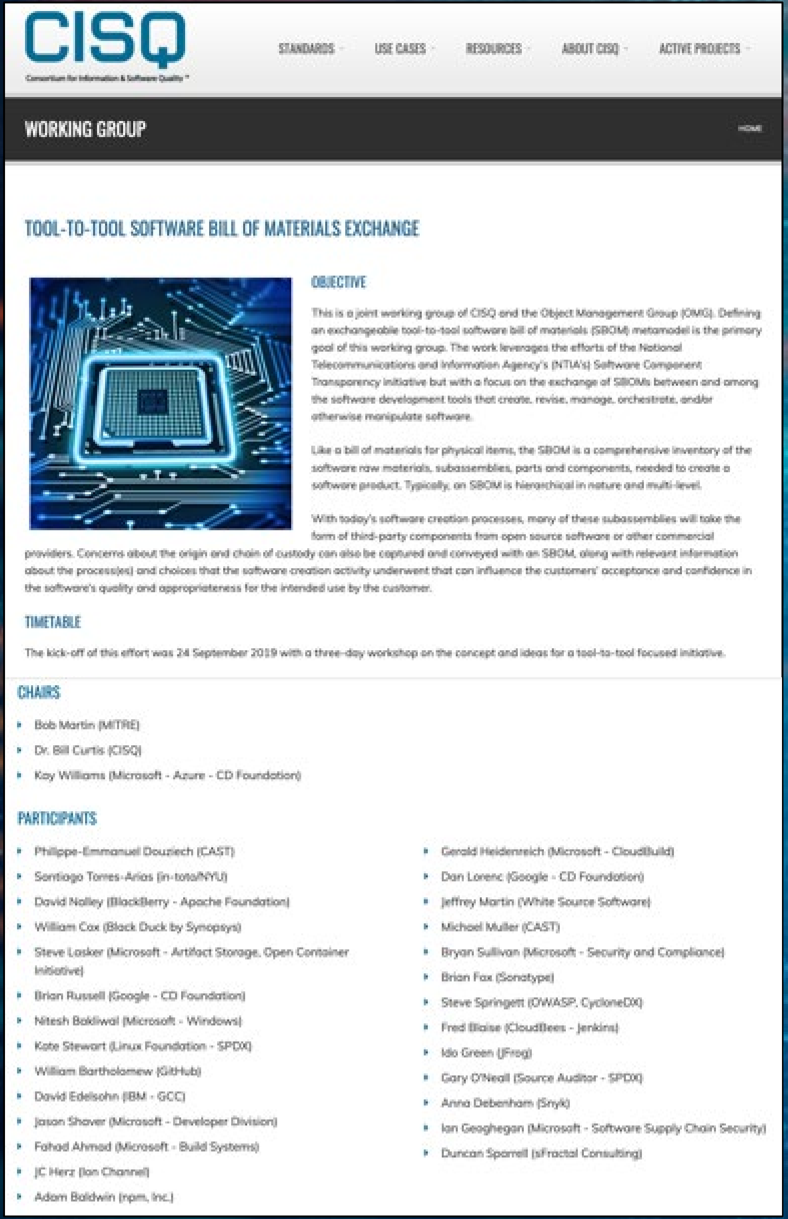
Item Hash/Signature

SBoM Hash/Signature

Correlated Info

Desired sequence of ordered software supply chain steps, and requirements for each step for a specific project of interest

Launched 24 Sep 2019



TOOL-TO-TOOL SOFTWARE BILL OF MATERIALS EXCHANGE

OBJECTIVE

This is a joint working group of CISQ and the Object Management Group (OMG). Defining an exchangeable tool-to-tool software bill of materials (SBOM) metamodel is the primary goal of this working group. The work leverages the efforts of the National Telecommunications and Information Agency's (NTIA's) Software Component Transparency initiative but with a focus on the exchange of SBOMs between and among the software development tools that create, revise, manage, orchestrate, and/or otherwise manipulate software.

Like a bill of materials for physical items, the SBOM is a comprehensive inventory of the software raw materials, subassemblies, parts and components, needed to create a software product. Typically, an SBOM is hierarchical in nature and multi-level.

With today's software creation processes, many of these subassemblies will take the form of third-party components from open source software or other commercial providers. Concerns about the origin and chain of custody can also be captured and conveyed with an SBOM, along with relevant information about the process(es) and choices that the software creation activity underwent that can influence the customers' acceptance and confidence in the software's quality and appropriateness for the intended use by the customer.

TIMETABLE

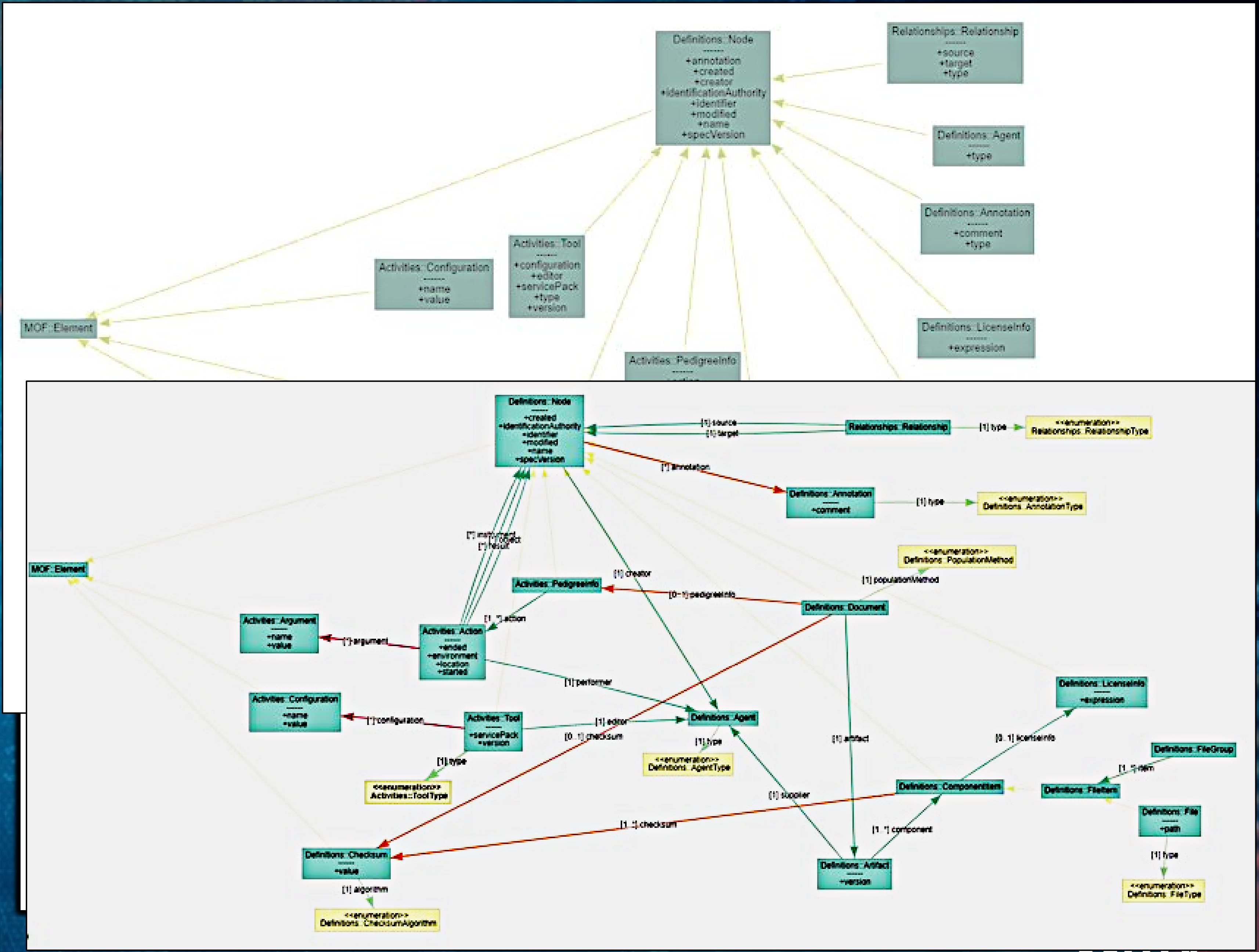
The kick-off of this effort was 24 September 2019 with a three-day workshop on the concept and ideas for a tool-to-tool focused initiative.

CHAIRS

- Bob Martin (MITRE)
- Dr. Bill Curtis (CISQ)
- Kay Williams (Microsoft - Azure - CD Foundation)

PARTICIPANTS

- Philippe-Emmanuel Douziech (CAST)
- Santiago Torres-Arias (in-toto/NYU)
- David Nalley (BlackBerry - Apache Foundation)
- William Cox (Black Duck by Synopsys)
- Steve Lasker (Microsoft - Artifact Storage, Open Container Initiative)
- Brian Russell (Google - CD Foundation)
- Nitesh Bakliwal (Microsoft - Windows)
- Kate Stewart (Linux Foundation - SPDX)
- William Bartholomew (GitHub)
- David Edelsohn (IBM - GCC)
- Jason Shaver (Microsoft - Developer Division)
- Fahad Ahmad (Microsoft - Build Systems)
- JC Herz (Ion Channel)
- Adam Baldwin (npm, Inc.)
- Gerald Heidenreich (Microsoft - CloudBuild)
- Dan Lorenc (Google - CD Foundation)
- Jeffrey Martin (White Source Software)
- Michael Muller (CAST)
- Bryan Sullivan (Microsoft - Security and Compliance)
- Brian Fox (Sonatype)
- Steve Springett (OWASP, CycloneDX)
- Fred Blaise (CloudBees - Jenkins)
- Ido Green (JFrog)
- Gary O'Neall (Source Auditor - SPDX)
- Anna Debenham (Snyk)
- Ian Geoghegan (Microsoft - Software Supply Chain Security)
- Duncan Sparrell (sFractal Consulting)



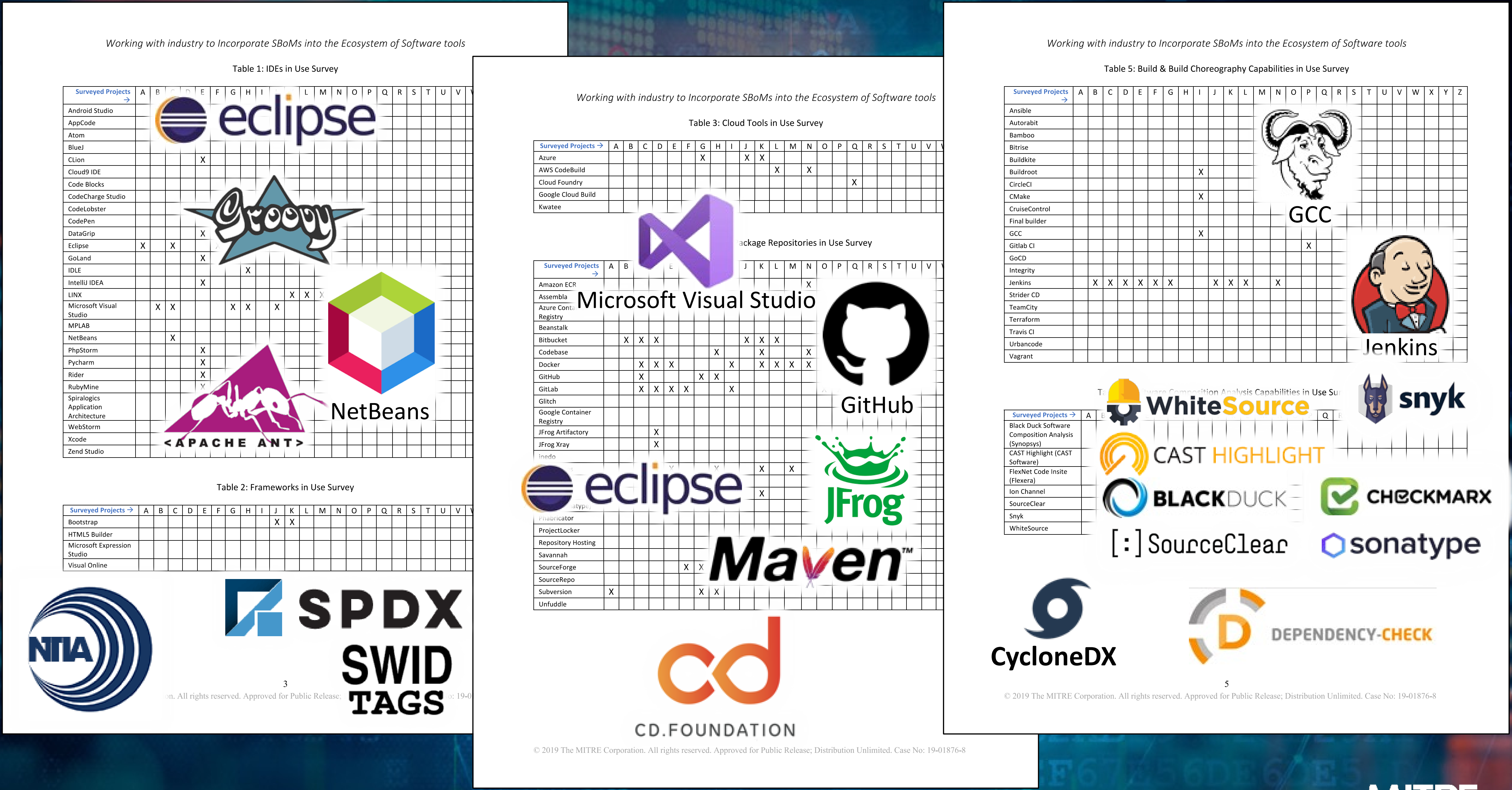


Table 1: IDEs in Use Survey

Surveyed Projects →	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
Android Studio																						
AppCode																						
Atom																						
BlueJ																						
CLion					X																	
Cloud9 IDE																						
Code Blocks																						
CodeCharge Studio																						
CodeLobster																						
CodePen																						
DataGrip					X																	
Eclipse	X		X			X																
GoLand					X																	
IDLE								X														
IntelliJ IDEA					X																	
LINX											X	X	X									
Microsoft Visual Studio		X	X				X	X		X												
MPLAB																						
NetBeans			X																			
PhpStorm					X																	
Pycharm					X																	
Rider					X																	
RubyMine					X																	
Spiralogics Application Architecture																						
WebStorm																						
Xcode																						
Zend Studio																						

Table 2: Frameworks in Use Survey

Surveyed Projects →	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
Bootstrap										X	X											
HTML5 Builder																						
Microsoft Expression Studio																						
Visual Online																						

Table 3: Cloud Tools in Use Survey

Surveyed Projects →	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
Azure							X			X	X											
AWS CodeBuild												X		X								
Cloud Foundry																	X					
Google Cloud Build																						
Kwatee																						

ackage Repositories in Use Survey

Surveyed Projects →	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
Amazon ECR														X								
Assembla																						
Azure Conta Registry																						
Beanstalk																						
Bitbucket		X	X	X						X	X	X										
Codebase								X			X			X								
Docker			X	X	X				X		X	X	X	X								
GitHub			X				X	X														
GitLab			X	X	X	X			X						X	X						
Glitch																						
Google Container Registry																						
JFrog Artifactory				X																		
JFrog Xray				X																		
inedo																						
[illegible]					X			X			X		X									
[illegible]																						
[illegible]											X											
natype)																						
Phabricator																						
ProjectLocker																						
Repository Hosting																						
Savannah																						
SourceForge						X	X															
SourceRepo																						
Subversion	X						X	X														
Unfuddle																						

Table 5: Build & Build Choreography Capabilities in Use Survey

Surveyed Projects →	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ansible																										
Autorabit																										
Bamboo																										
Bitrise																										
Buildkite																										
Buildroot									X																	
CircleCI																										
CMake									X																	
CruiseControl																										
Final builder																										
GCC									X																	
Gitlab CI																X										
GoCD																										
Integrity																										
Jenkins		X	X	X	X	X	X			X	X	X		X												
Strider CD																										
TeamCity																										
Terraform																										
Travis CI																										
Urbancode																										
Vagrant																										

Ta ware Composition Analysis Capabilities in Use Sur

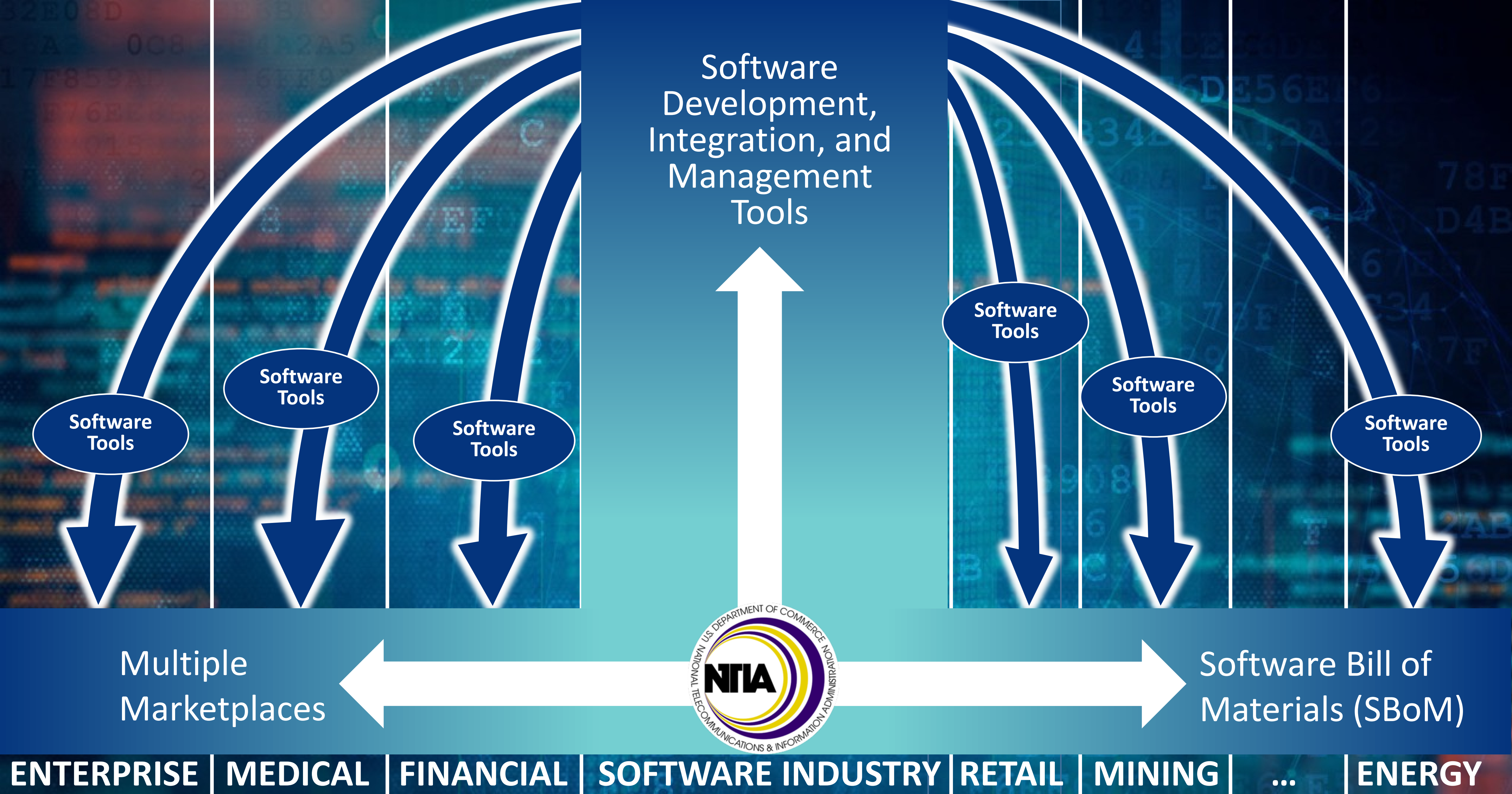
Surveyed Projects →	A	B			E	F	G	H	I	J	K	L	M	N	O	P	Q	R
Black Duck Software Composition Analysis (Synopsys)																		
CAST Highlight (CAST Software)																		
FlexNet Code Insite (Flexera)																		
Ion Channel																		
SourceClear																		
Snyk																		
WhiteSource																		

MITRE

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

MITRE



Software Development, Integration, and Management Tools

Software Tools

Software Tools

Software Tools

Software Tools

Software Tools

Software Tools

Multiple Marketplaces

Software Bill of Materials (SBoM)

U.S. DEPARTMENT OF COMMERCE
NATIONAL TELECOMMUNICATIONS & INFORMATION ADMINISTRATION
NTIA

ENTERPRISE | MEDICAL | FINANCIAL | SOFTWARE INDUSTRY | RETAIL | MINING | … | ENERGY

MITRE

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

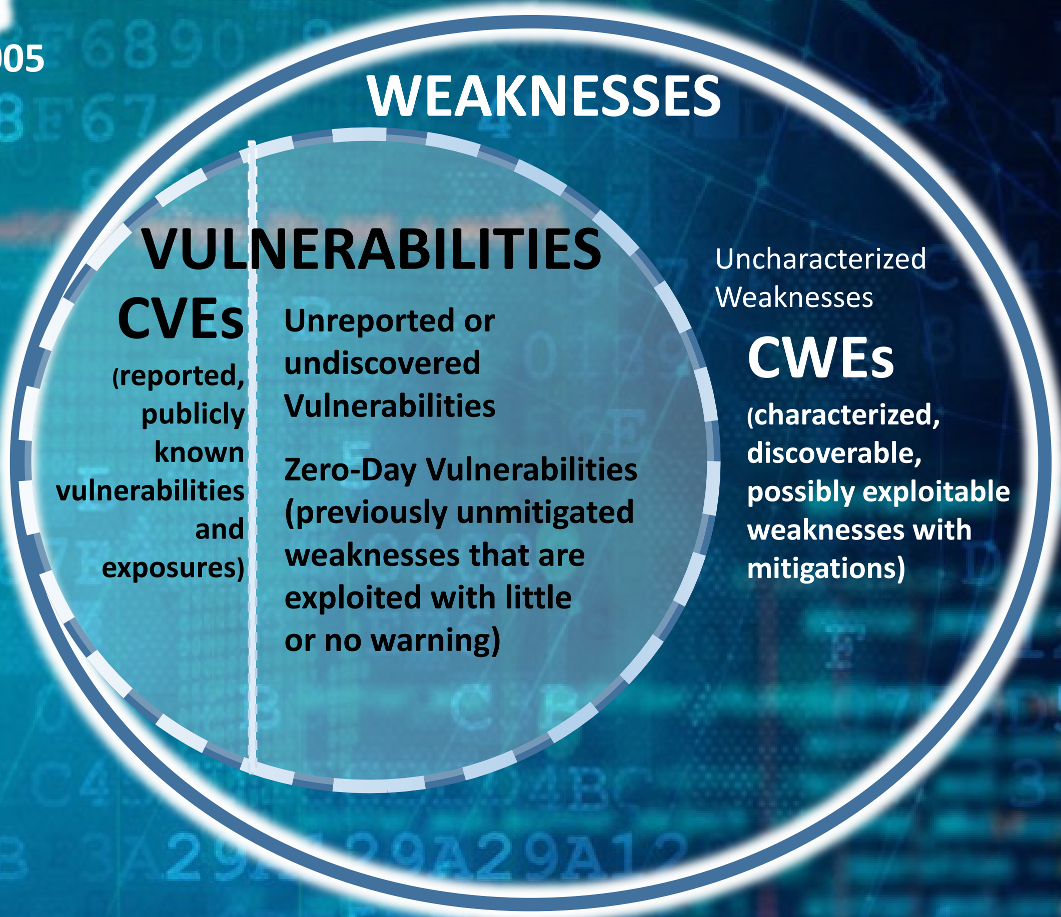
CWE 2005

CWSS 2012

CVE 1999

CVSS 2002

CAPEC 2005



WEAKNESSES

VULNERABILITIES

CVEs (reported, publicly known vulnerabilities and exposures)

Unreported or undiscovered Vulnerabilities

Zero-Day Vulnerabilities (previously unmitigated weaknesses that are exploited with little or no warning)

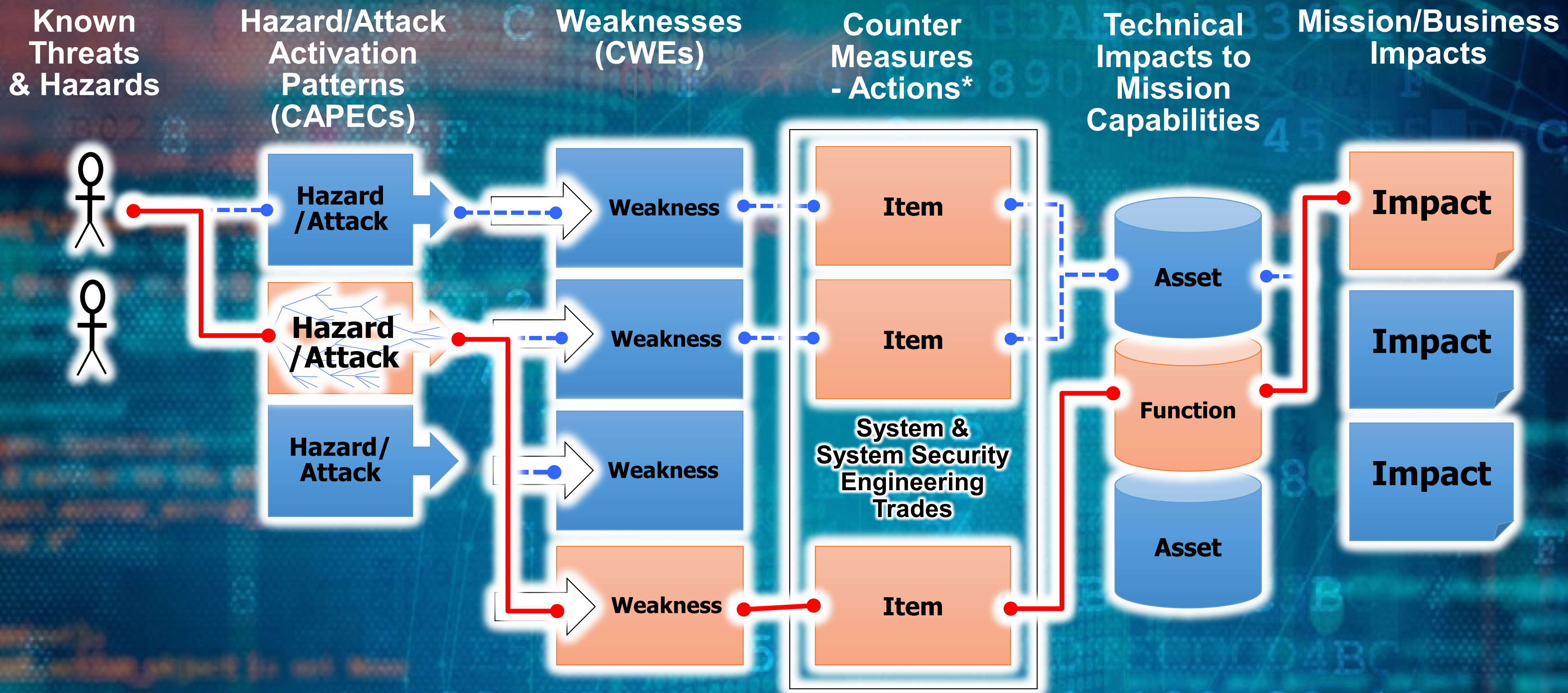
Uncharacterized Weaknesses

CWEs (characterized, discoverable, possibly exploitable weaknesses with mitigations)

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.

MITRE

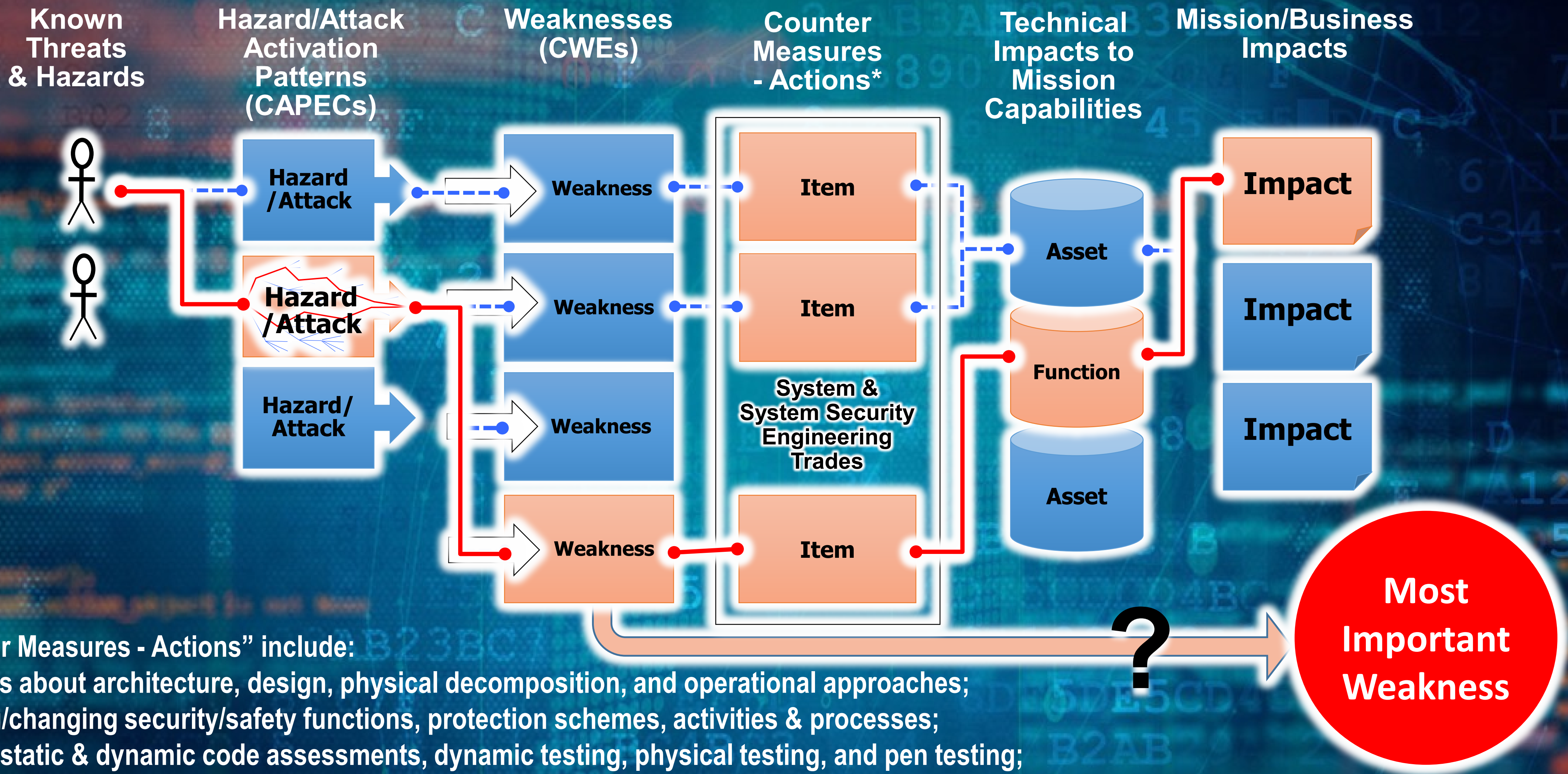
Assurance needs to address the Hazards & Attacks that can impact SW-Based Mission Functions



"Counter Measures - Actions" include:

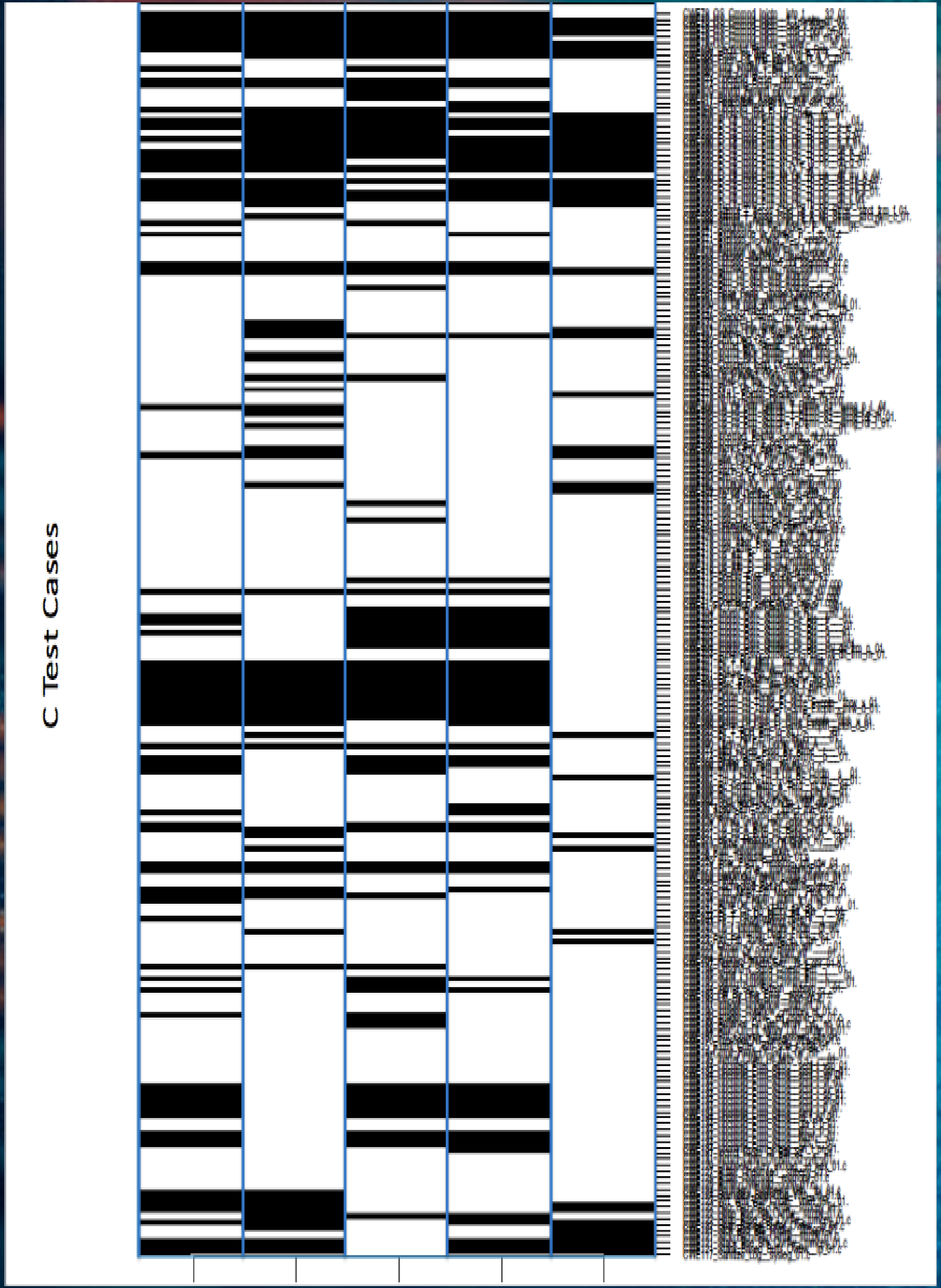
choices about architecture, design, physical decomposition, and operational approaches;
adding/changing security/safety functions, protection schemes, activities & processes;
use of static & dynamic code assessments, dynamic testing, physical testing, and pen testing;
attack surface & fault-tree analysis,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

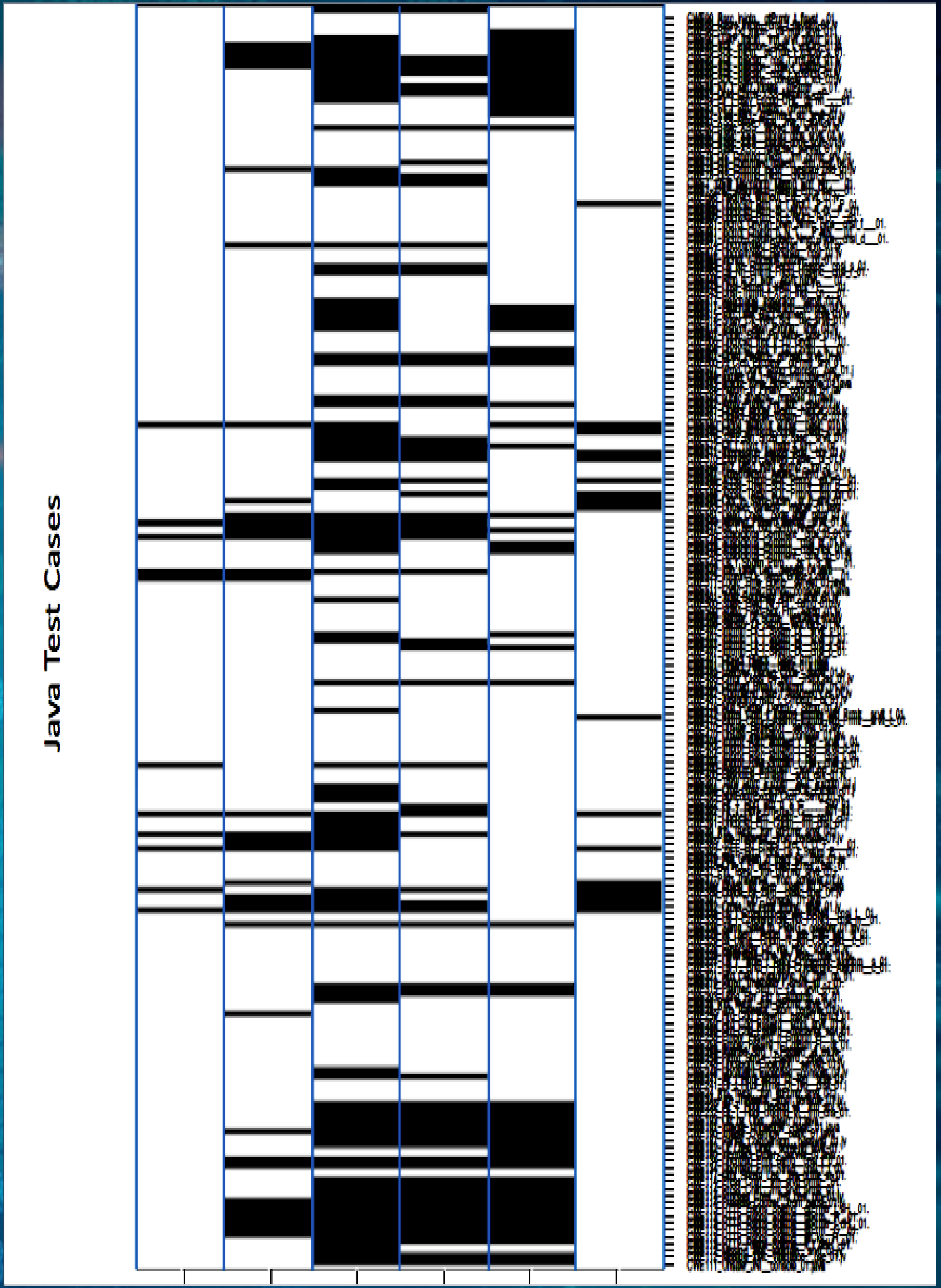
Assurance needs to address the Hazards & Attacks that can impact SW-Based Mission Functions



"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


C Test Cases


Java Test Cases

Utilizing Appropriate Detection Methods to Collect Evidence to Gain Assurance…

Artifacts

CONOPS

Requirements

Architecture

Design

Process

Code

Binary

Running Binary

Environment of System

Use of Mission Software

Detection Methods

Design Review

Code Review

Attack Surface Analysis

Static Analysis Tool A

Static Analysis Tool B

Dynamic Analysis Tool C

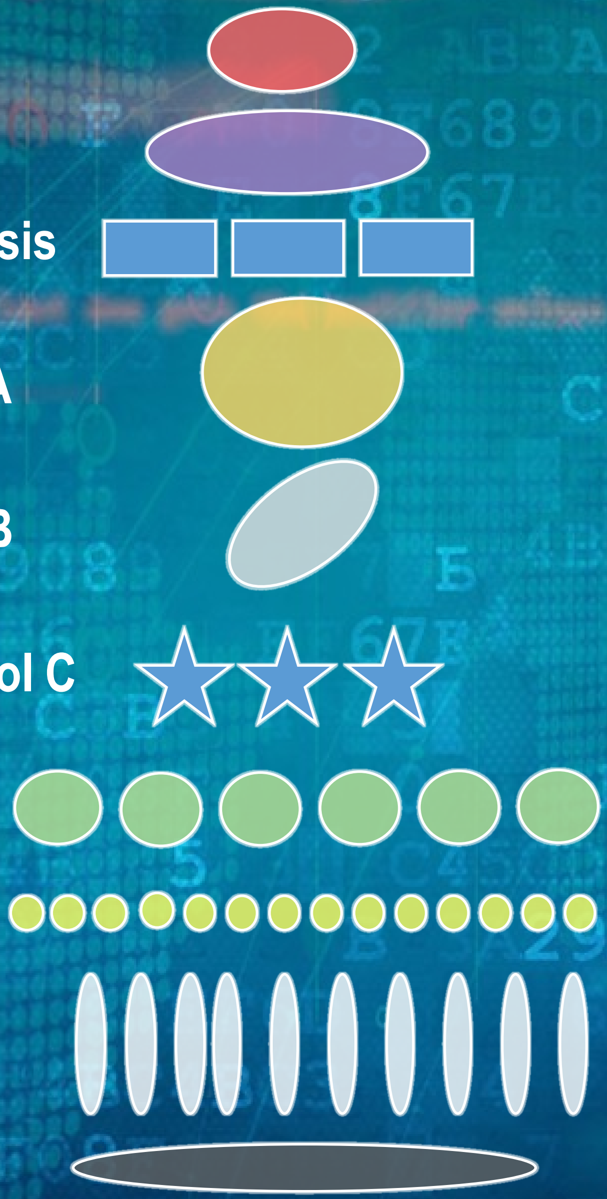
Fuzz Testing

Pen Testing

Blue Teaming

Red Teaming

Coverage

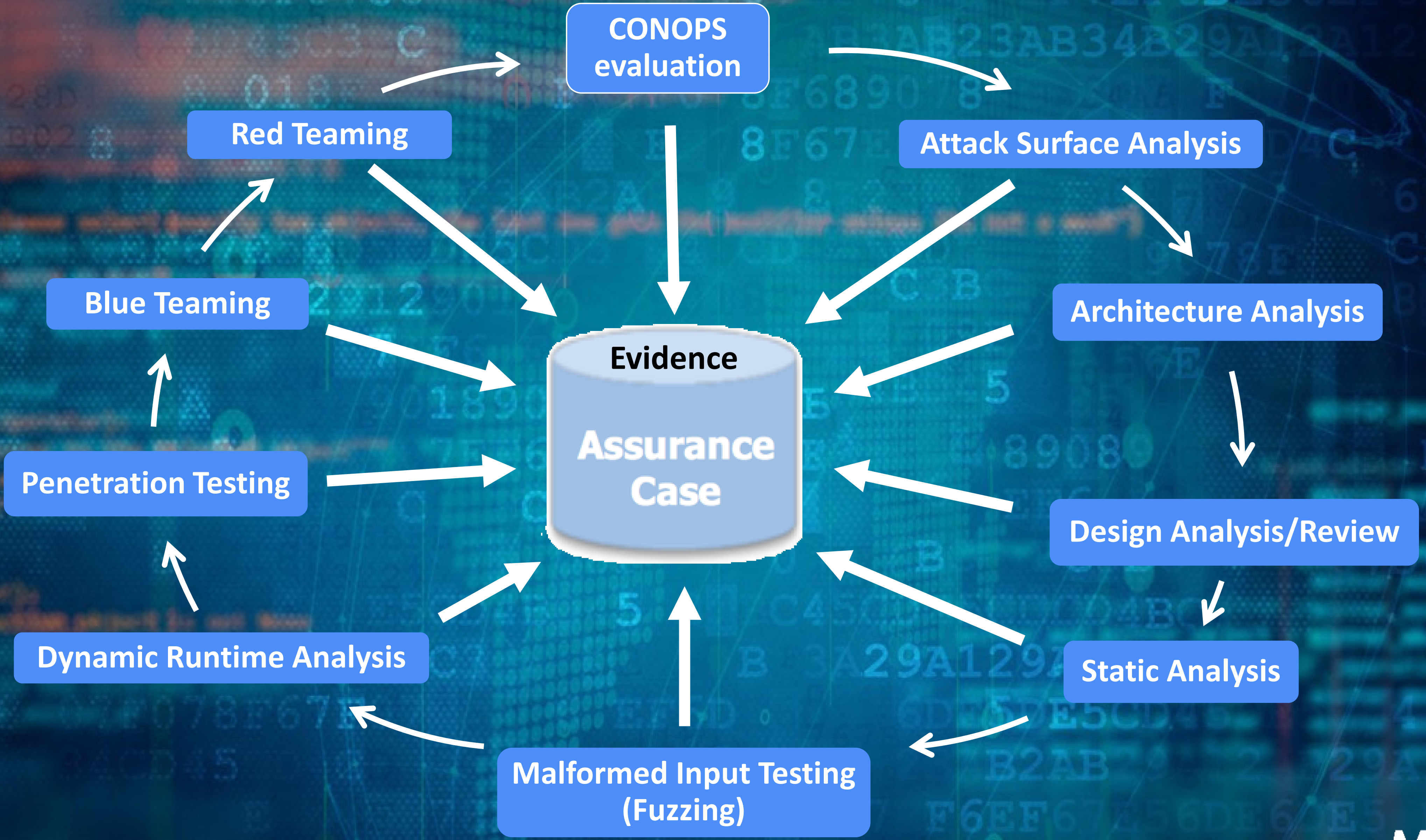


CVE,
CWE,
CAPEC, …

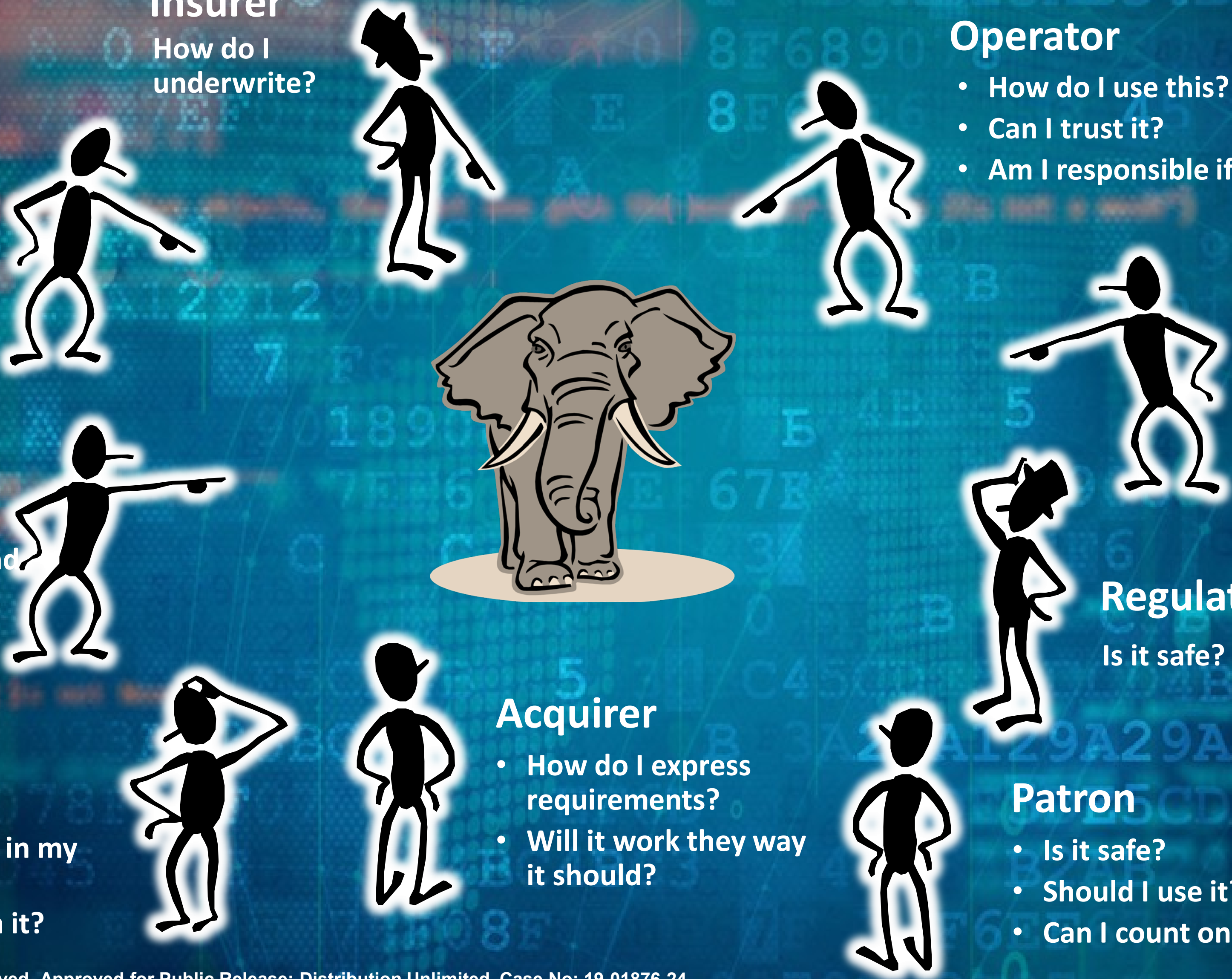
Most Important Quality Issues

MITRE

Multiple Sources of Assurance Evidence from Throughout the Lifecycle of the item(s) needing Assurance.



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?

Acquirer

- How do I express requirements?
- Will it work they way it should?

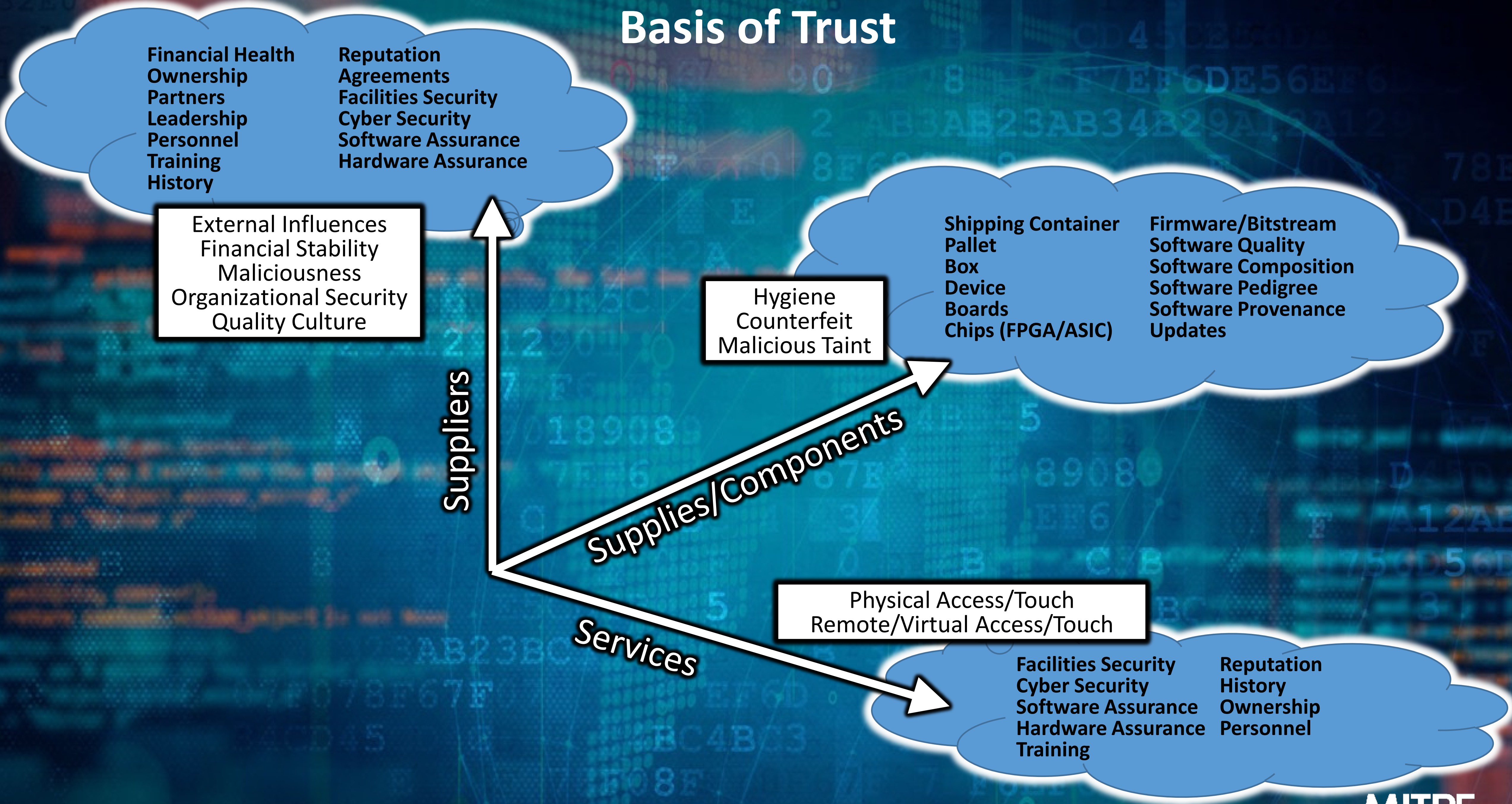
Community

- 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?

Basis of Trust



Financial Health
Ownership
Partners
Leadership
Personnel
Training
History

Reputation
Agreements
Facilities Security
Cyber Security
Software Assurance
Hardware Assurance

External Influences
Financial Stability
Maliciousness
Organizational Security
Quality Culture

Suppliers

Hygiene
Counterfeit
Malicious Taint

Shipping Container
Pallet
Box
Device
Boards
Chips (FPGA/ASIC)

Firmware/Bitstream
Software Quality
Software Composition
Software Pedigree
Software Provenance
Updates

Supplies/Components

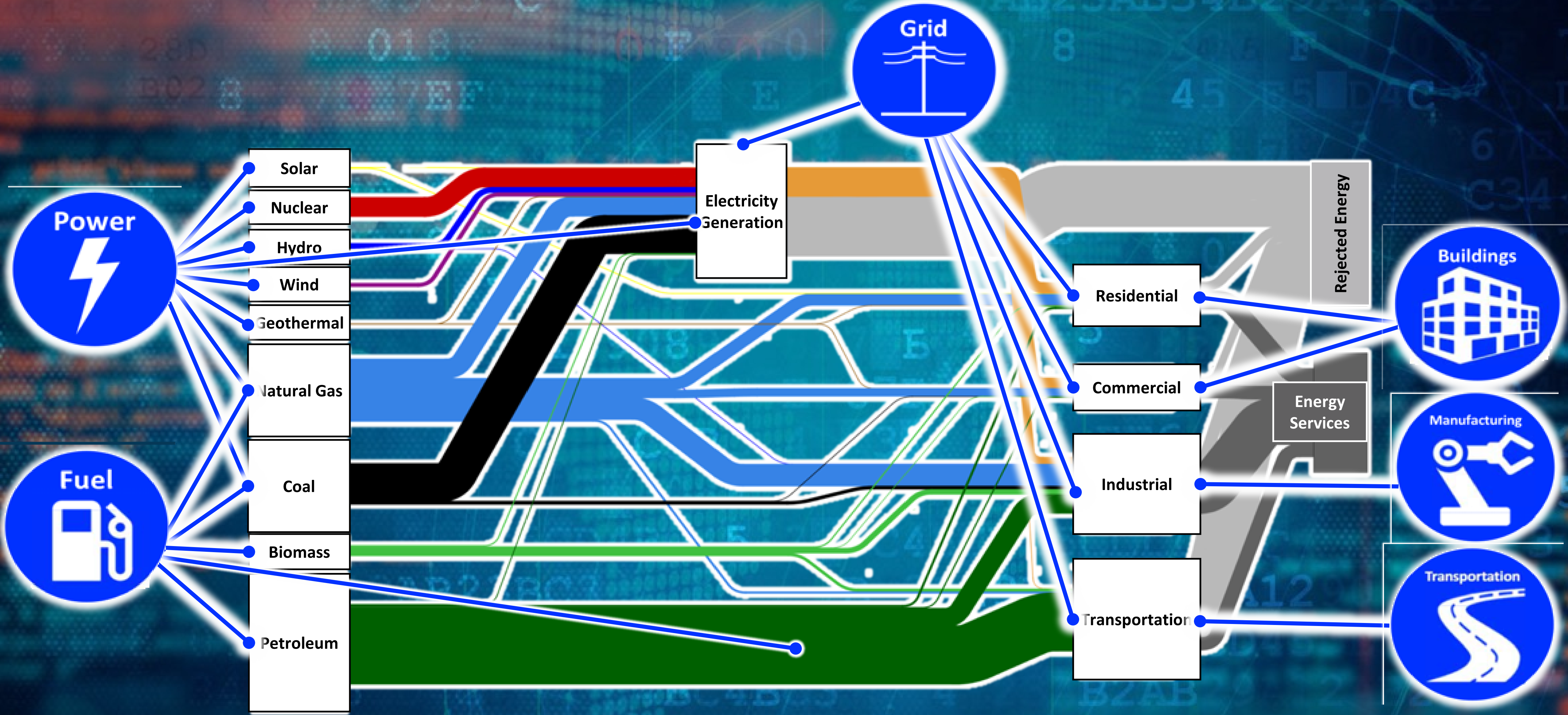
Services

Physical Access/Touch
Remote/Virtual Access/Touch

Facilities Security
Cyber Security
Software Assurance
Hardware Assurance
Training

Reputation
History
Ownership
Personnel

Need Standards to Drive Consistency in Discussing and Conveying Assurance due to the Sector-2-Sector linkages



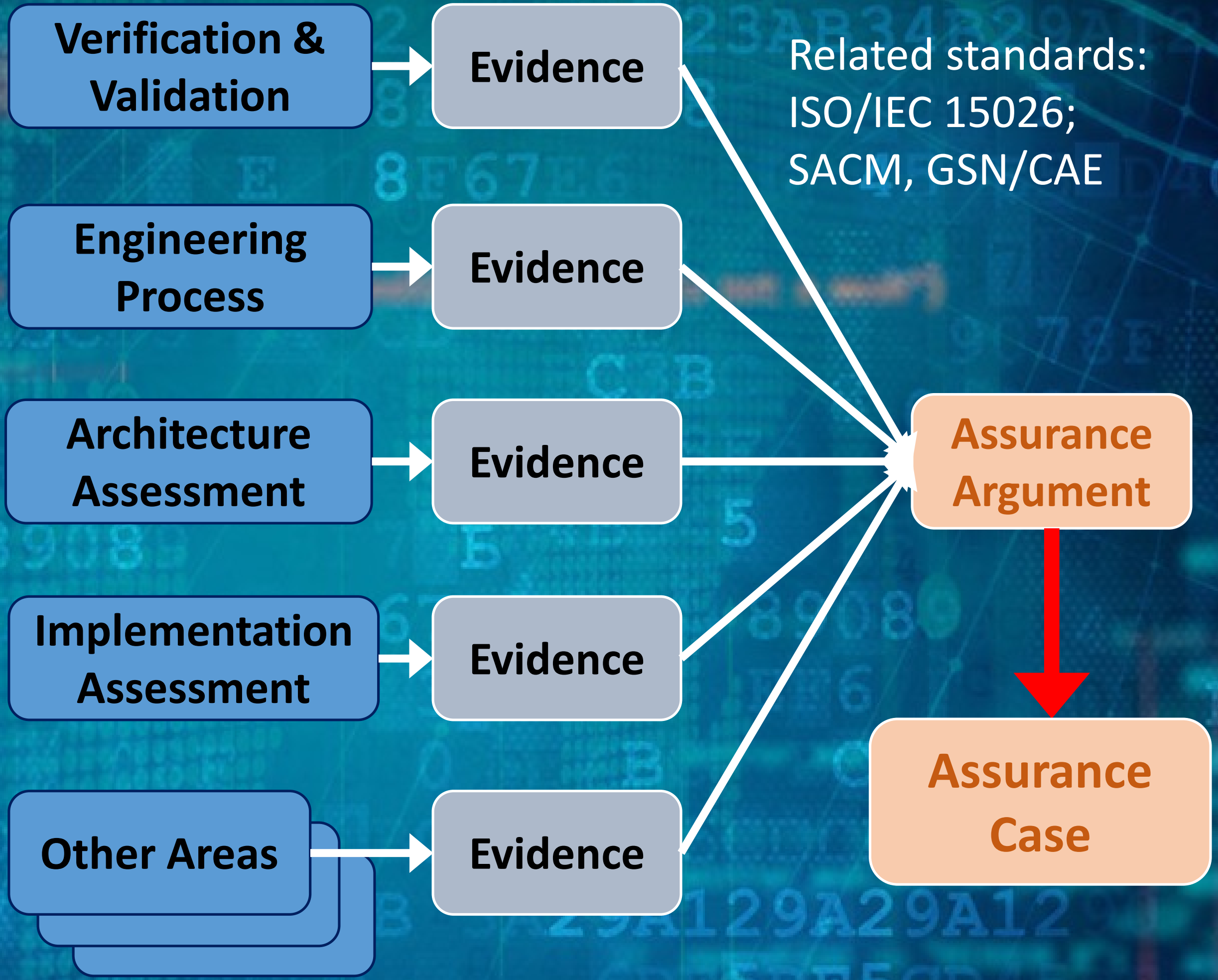
MITRE

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



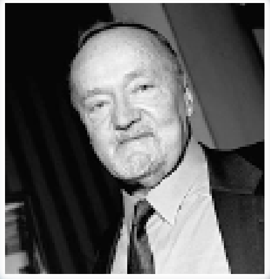
Related standards:
ISO/IEC 15026;
SACM, GSN/CAE

TYPES OF EVIDENCE FOR AN ASSURANCE CASE

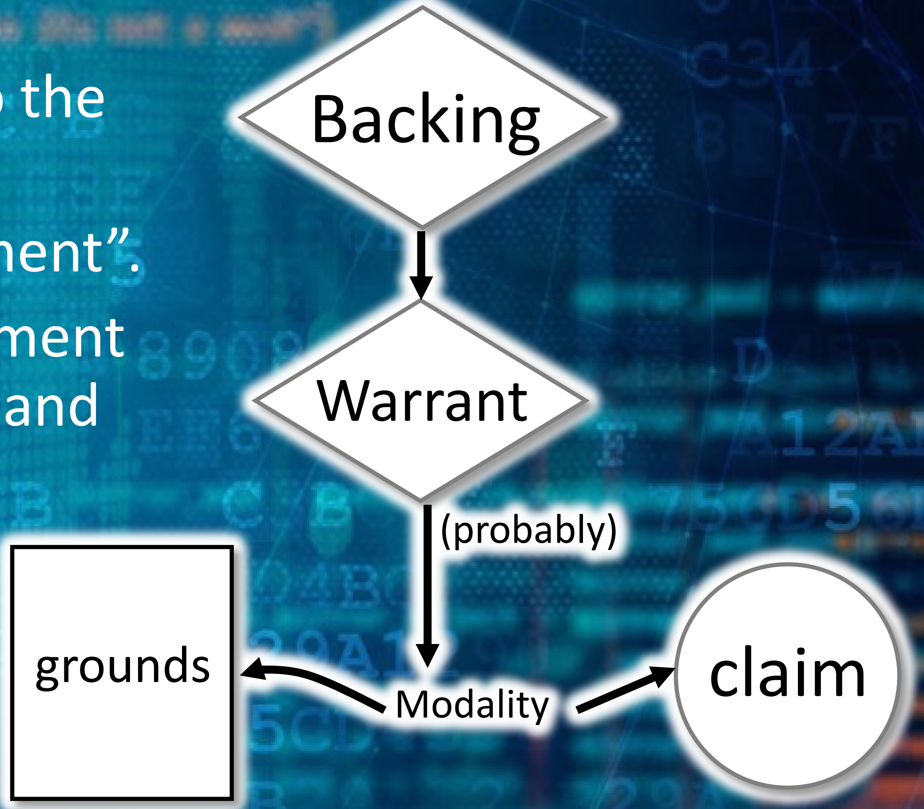
Confidence demands objectivity, scientific method and cost-effectiveness

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.



Stephen Toulmin, 1958

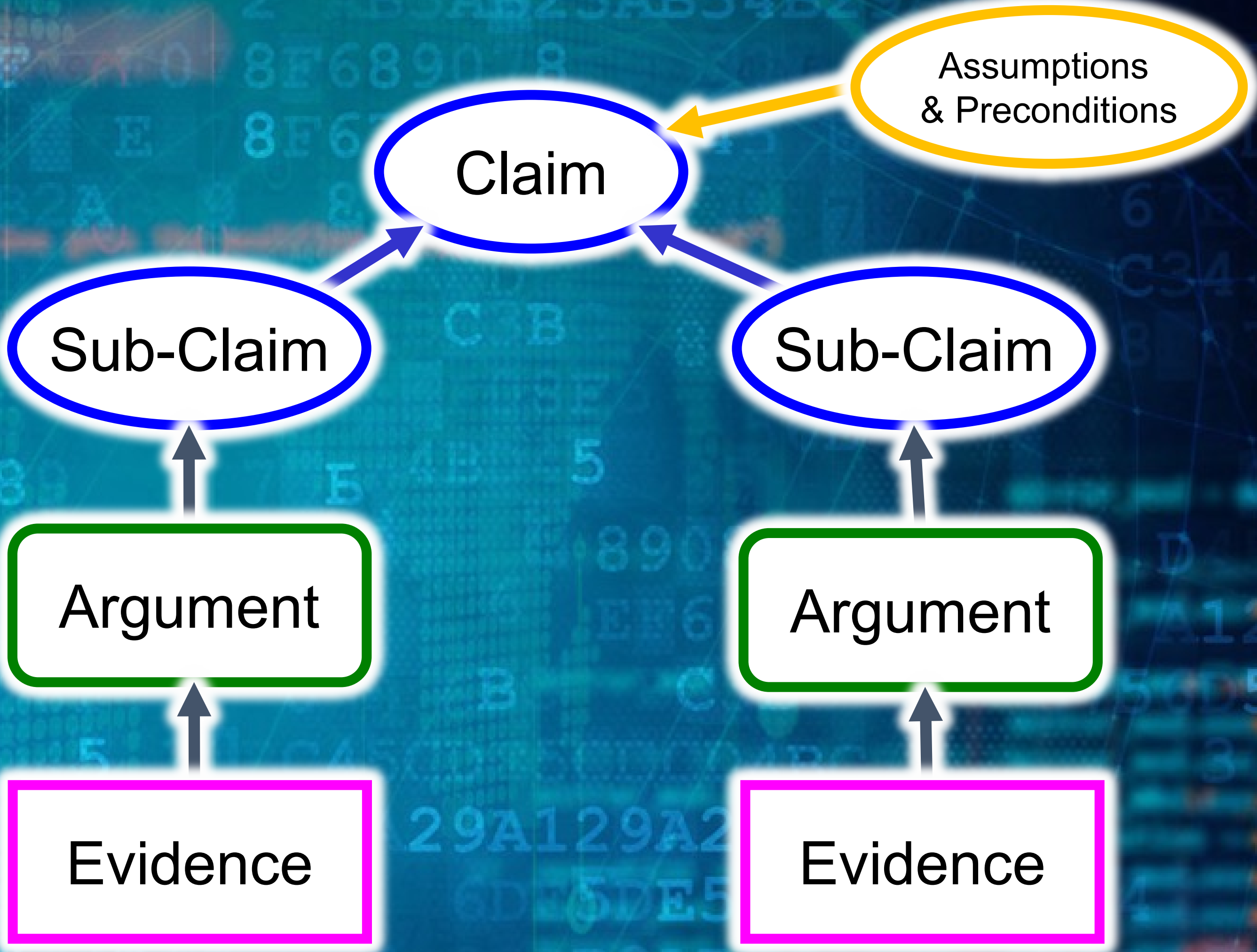


The Basics of an Assurance Case

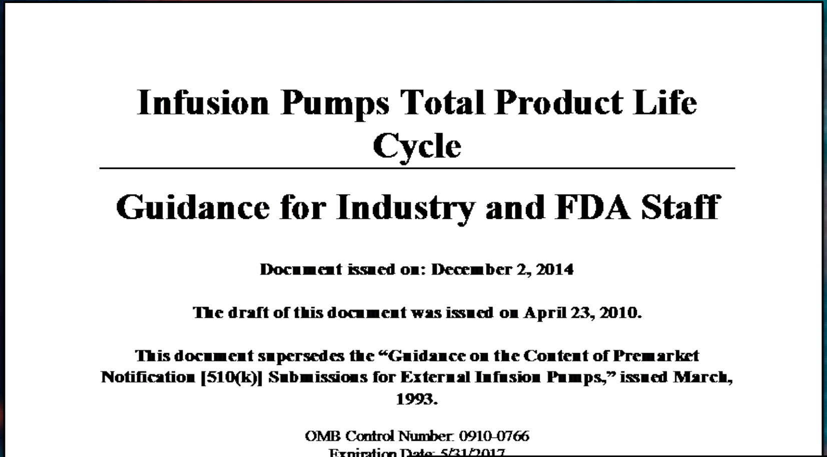
Claim =
assertion to be proven

Argument =
how evidence supports claim

Evidence =
required documentation



MITRE



Infusion Pumps Total Product Life Cycle

Guidance for Industry and FDA Staff

Document issued on: December 2, 2014

The draft of this document was issued on April 23, 2010.

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

OMB Control Number: 0910-0766
Expiration Date: 5/31/2017

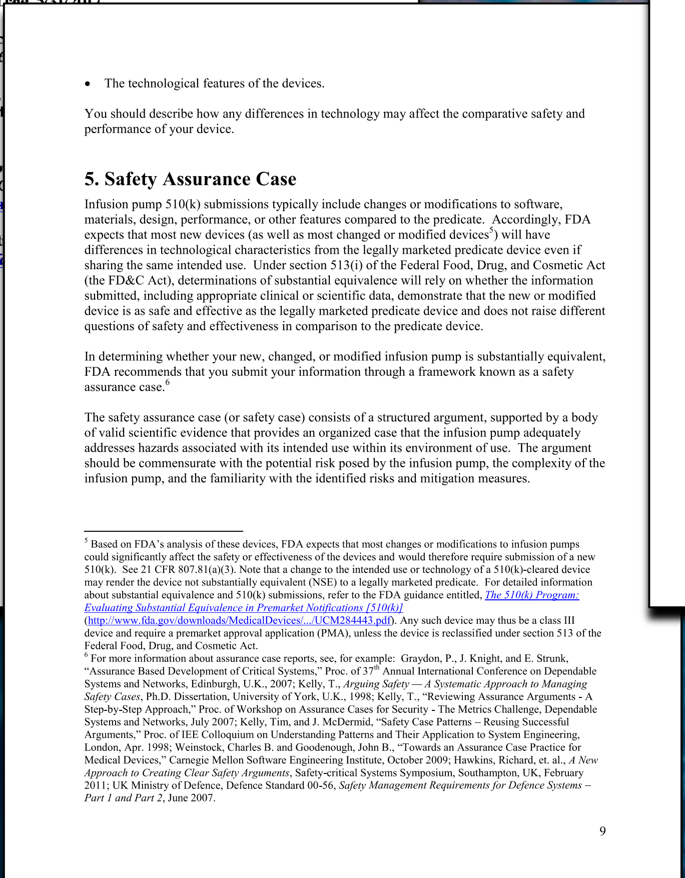
For questions regarding this document, please c
Branch, Office of Device Evaluation at 301-79

For questions regarding safety assurance cases,
Devices Branch, Office of Device Evaluation at
richard.chapman@fda.hhs.gov.

For questions regarding pre-clearance inspectio
Ear/Nose/Throat, General Hospital, Infectious C
Compliance at 301-796-5770 or via email at fra

For questions pertaining to manufacturer report
301-796-6104 or via email at sharon.kapsch@fd

Center for Devices and Radiological Health
CDRH



- The technological features of the devices.

You should describe how any differences in technology may affect the comparative safety and performance of your device.

5. Safety Assurance Case

Infusion pump 510(k) submissions typically include changes or modifications to software, materials, design, performance, or other features compared to the predicate. Accordingly, FDA expects that most new devices (as well as most changed or modified devices[5]) will have differences in technological characteristics from the legally marketed predicate device even if sharing the same intended use. Under section 513(i) of the Federal Food, Drug, and Cosmetic Act (the FD&C Act), determinations of substantial equivalence will rely on whether the information submitted, including appropriate clinical or scientific data, demonstrate that the new or modified device is as safe and effective as the legally marketed predicate device and does not raise different questions of safety and effectiveness in comparison to the predicate device.

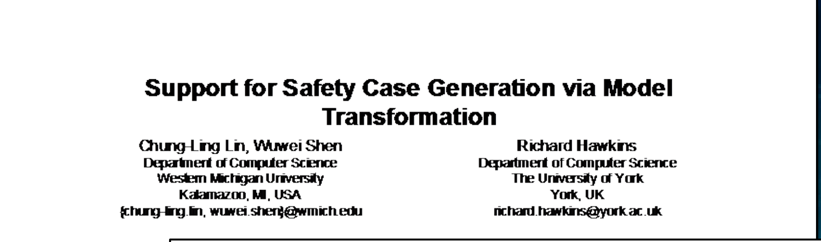
In determining whether your new, changed, or modified infusion pump is substantially equivalent, FDA recommends that you submit your information through a framework known as a safety assurance case.[6]

The safety assurance case (or safety case) consists of a structured argument, supported by a body of valid scientific evidence that provides an organized case that the infusion pump adequately addresses hazards associated with its intended use within its environment of use. The argument should be commensurate with the potential risk posed by the infusion pump, the complexity of the infusion pump, and the familiarity with the identified risks and mitigation measures.

[5] Based on FDA's analysis of these devices, FDA expects that most changes or modifications to infusion pumps could significantly affect the safety or effectiveness of the devices and would therefore require submission of a new 510(k). See 21 CFR 807.81(a)(3). Note that a change to the intended use or technology of a 510(k)-cleared device may render the device not substantially equivalent (NSE) to a legally marketed predicate. For detailed information about substantial equivalence and 510(k) submissions, refer to the FDA guidance entitled, *The 510(k) Program: Evaluating Substantial Equivalence in Premarket Notifications [510(k)]* (http://www.fda.gov/downloads/MedicalDevices/.../UCM284443.pdf). Any such device may thus be a class III device and require a premarket approval application (PMA), unless the device is reclassified under section 513 of the Federal Food, Drug, and Cosmetic Act.

[6] For more information about assurance case reports, see, for example: Graydon, P., J. Knight, and E. Strunk, "Assurance Based Development of Critical Systems," Proc. of 37th Annual International Conference on Dependable Systems and Networks, Edinburgh, U.K., 2007; Kelly, T., *Arguing Safety — A Systematic Approach to Managing Safety Cases*, Ph.D. Dissertation, University of York, U.K., 1998; Kelly, T., "Reviewing Assurance Arguments - A Step-by-Step Approach," Proc. of Workshop on Assurance Cases for Security - The Metrics Challenge, Dependable Systems and Networks, July 2007; Kelly, Tim, and J. McDermid, "Safety Case Patterns – Reusing Successful Arguments," Proc. of IEE Colloquium on Understanding Patterns and Their Application to System Engineering, London, Apr. 1998; Weinstock, Charles B. and Goodenough, John B., "Towards an Assurance Case Practice for Medical Devices," Carnegie Mellon Software Engineering Institute, October 2009; Hawkins, Richard, et. al., *A New Approach to Creating Clear Safety Arguments*, Safety-critical Systems Symposium, Southampton, UK, February 2011; UK Ministry of Defence, Defence Standard 00-56, *Safety Management Requirements for Defence Systems – Part 1 and Part 2*, June 2007.

9



Support for Safety Case Generation via Model Transformation

Chung-Ling Lin, Wuwei Shen
Department of Computer Science
Western Michigan University
Kalamazoo, MI, USA
{chung-ling.lin, wuwei.shen}@wmich.edu

Richard Hawkins
Department of Computer Science
The University of York
York, UK
richard.hawkins@york.ac.uk

ABSTRACT
Assessing the safe
systems under ever
confidence is a gr
alike. One method
the use of assuranc
little or too much a
affect confidence
automatic generatio
expedite a develop
perform compliance
framework which
metamodel, and a
generate a safety a
conformance of the
use the GPCA infus
this framework can
pump guidance publ

Keywords
Compliance check
systems; safety case

1. INTRODU
Assessing the safe
systems, such as
constraints with an
challenge for indust
to address this is
safety case in short
Administration (FD
guidance documen
pumps [2], which
use safety assurance
organize and presen
claims of their infu
infusion pump gu
automatically const

The construction an
system are a drau

Copyright retaine

SIGBED Rev

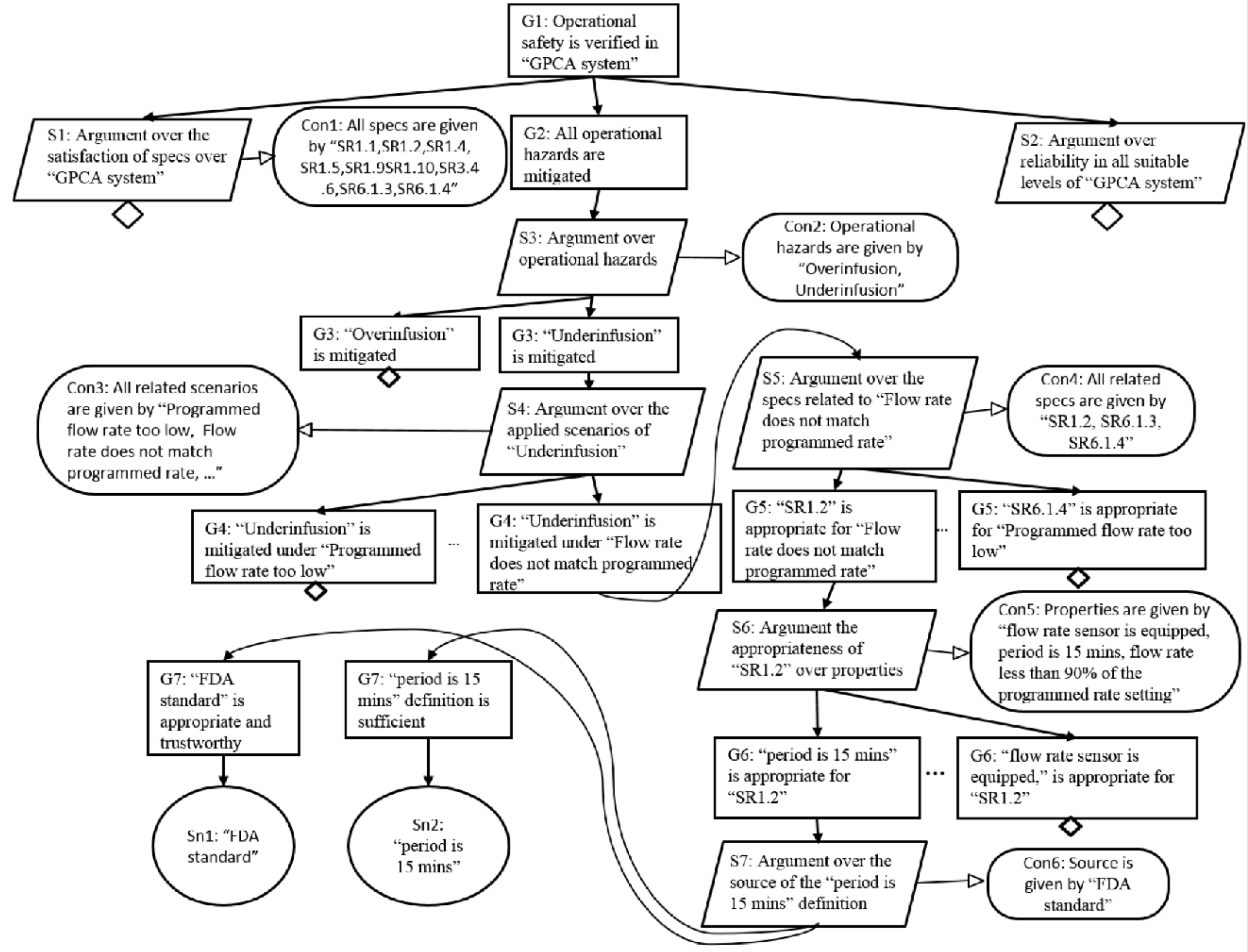

G1: Operational safety is verified in "GPCA system"
S1: Argument over the satisfaction of specs over "GPCA system"
Con1: All specs are given by "SR1.1,SR1.2,SR1.4, SR1.5,SR1.9SR1.10,SR3.4 .6,SR6.1.3,SR6.1.4"
G2: All operational hazards are mitigated
S2: Argument over reliability in all suitable levels of "GPCA system"
S3: Argument over operational hazards
Con2: Operational hazards are given by "Overinfusion, Underinfusion"
G3: "Overinfusion" is mitigated
G3: "Underinfusion" is mitigated
Con3: All related scenarios are given by "Programmed flow rate too low, Flow rate does not match programmed rate, ..."
S4: Argument over the applied scenarios of "Underinfusion"
S5: Argument over the specs related to "Flow rate does not match programmed rate"
Con4: All related specs are given by "SR1.2, SR6.1.3, SR6.1.4"
G4: "Underinfusion" is mitigated under "Programmed flow rate too low"
...
G4: "Underinfusion" is mitigated under "Flow rate does not match programmed rate"
G5: "SR1.2" is appropriate for "Flow rate does not match programmed rate"
...
G5: "SR6.1.4" is appropriate for "Programmed flow rate too low"
S6: Argument the appropriateness of "SR1.2" over properties
Con5: Properties are given by "flow rate sensor is equipped, period is 15 mins, flow rate less than 90% of the programmed rate setting"
G7: "FDA standard" is appropriate and trustworthy
G7: "period is 15 mins" definition is sufficient
G6: "period is 15 mins" is appropriate for "SR1.2"
...
G6: "flow rate sensor is equipped," is appropriate for "SR1.2"
Sn1: "FDA standard"
Sn2: "period is 15 mins"
S7: Argument over the source of the "period is 15 mins" definition
Con6: Source is given by "FDA standard"

Figure 9 Safety case model of GPCA system

The Assurance Case

THE Open GROUP

OMG OBJECT MANAGEMENT GROUP®

industrial internet CONSORTIUM

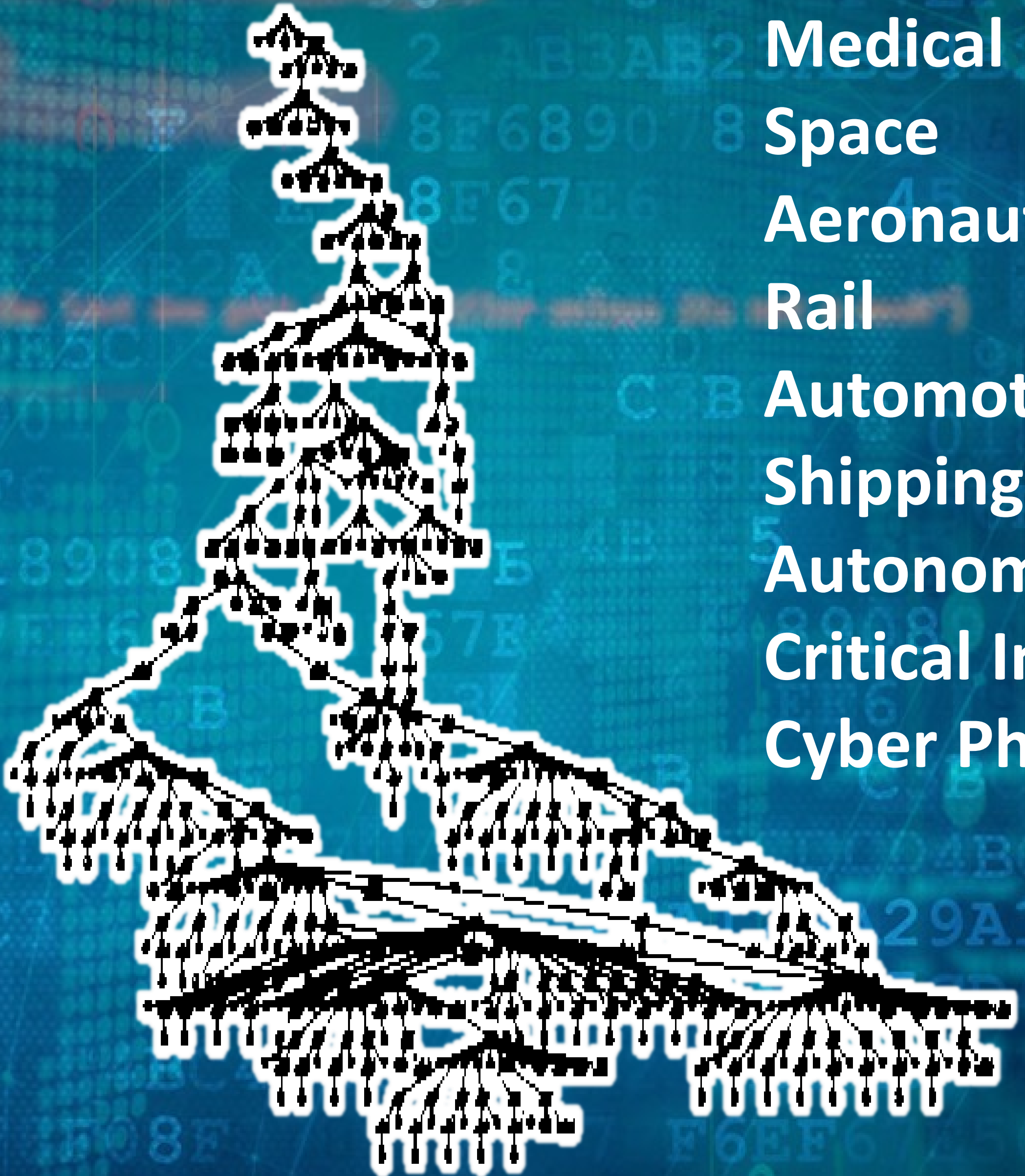
NASA

FDA

NIST

CITADEL
CRITICAL INFRASTRUCTURE PROTECTION USING ADAPTIVE MILS

DEIS
Dependability
Engineering
Innovation for Cyber Physical
Systems

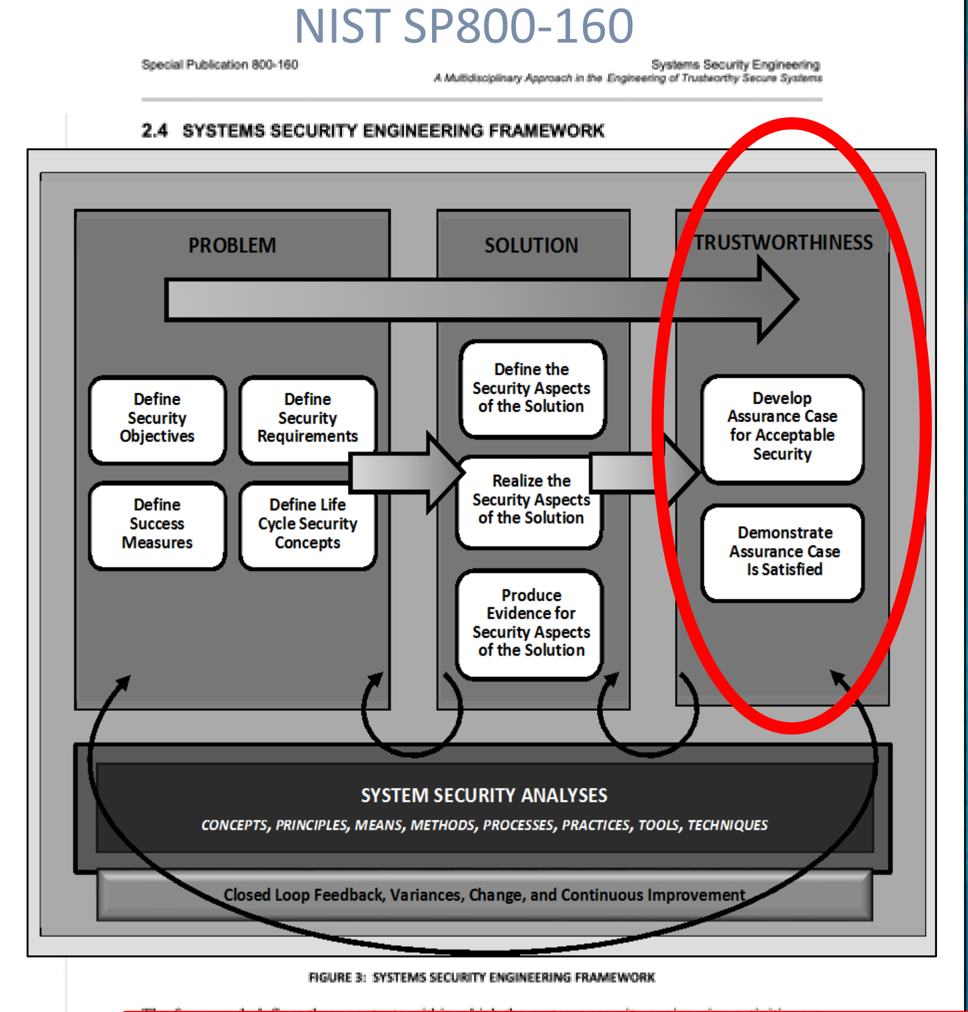


Medical
Space
Aeronautics
Rail
Automotive
Shipping
Autonomous
Critical Infrastructure
Cyber Physical Systems…

MITRE

Communicating Assurance to Gain Trust

NIST SP800-160



Special Publication 800-160

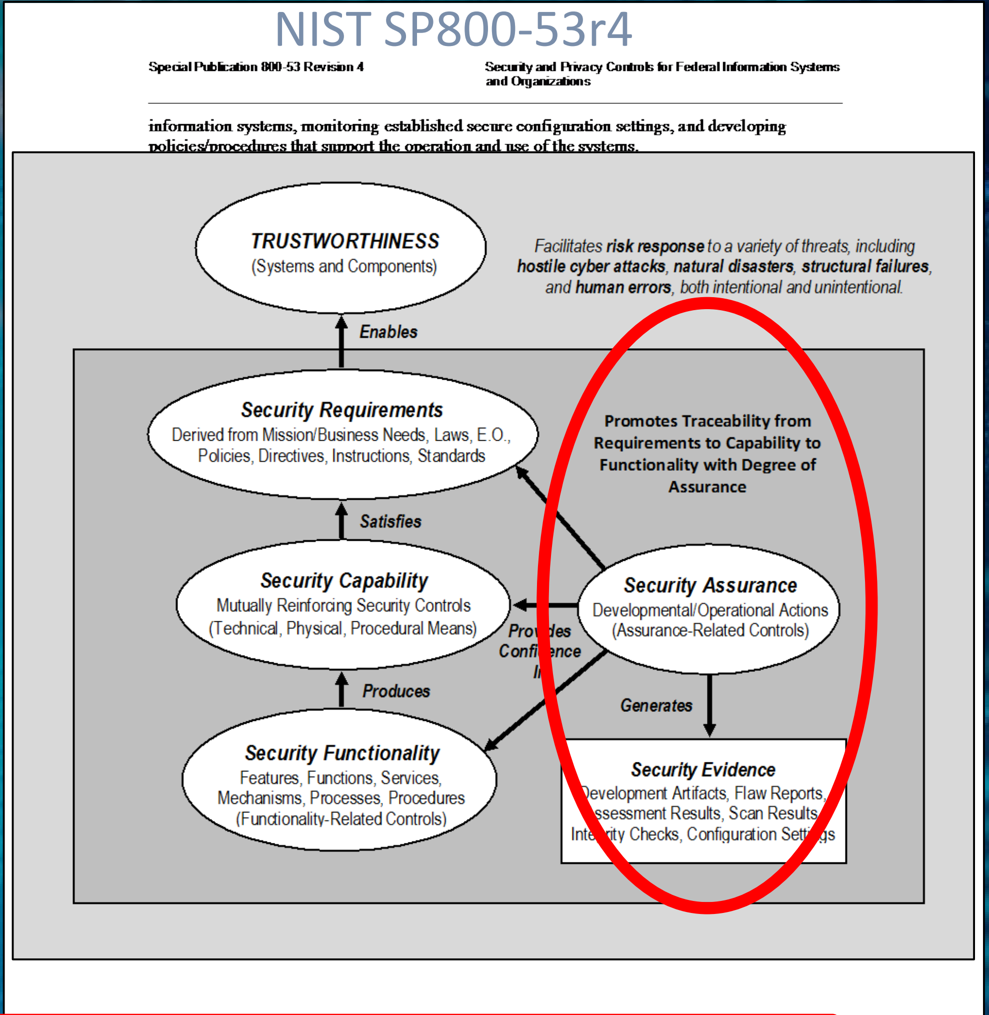
Systems Security Engineering
A Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems

2.4 SYSTEMS SECURITY ENGINEERING FRAMEWORK

PROBLEM
SOLUTION
TRUSTWORTHINESS
Define Security Objectives
Define Security Requirements
Define Success Measures
Define Life Cycle Security Concepts
Define the Security Aspects of the Solution
Realize the Security Aspects of the Solution
Produce Evidence for Security Aspects of the Solution
Develop Assurance Case for Acceptable Security
Demonstrate Assurance Case Is Satisfied
SYSTEM SECURITY ANALYSES
CONCEPTS, PRINCIPLES, MEANS, METHODS, PROCESSES, PRACTICES, TOOLS, TECHNIQUES
Closed Loop Feedback, Variances, Change, and Continuous Improvement

FIGURE 3: SYSTEMS SECURITY ENGINEERING FRAMEWORK

NIST SP800-53r4



Special Publication 800-53 Revision 4

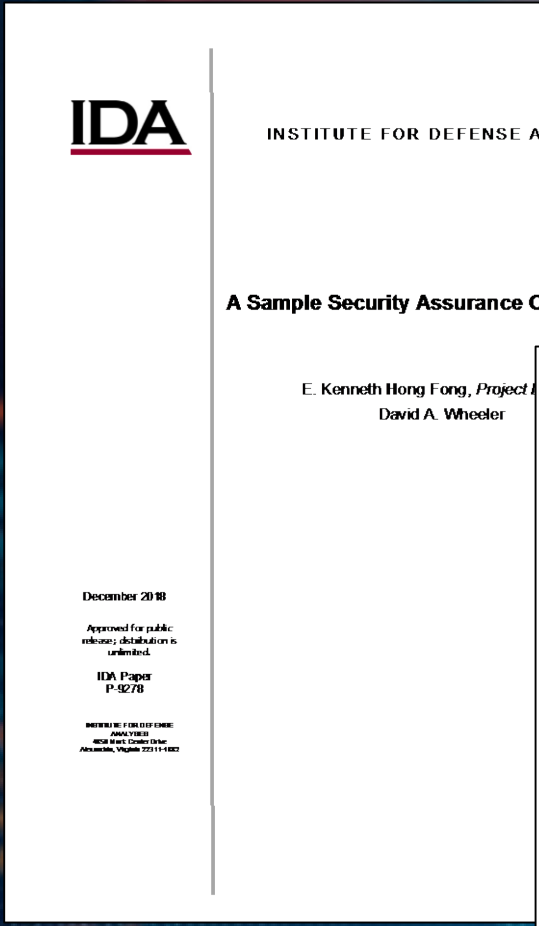
Security and Privacy Controls for Federal Information Systems and Organizations

information systems, monitoring established secure configuration settings, and developing policies/procedures that support the operation and use of the systems.

TRUSTWORTHINESS
(Systems and Components)
Facilitates risk response to a variety of threats, including hostile cyber attacks, natural disasters, structural failures, and human errors, both intentional and unintentional.
Enables
Security Requirements
Derived from Mission/Business Needs, Laws, E.O., Policies, Directives, Instructions, Standards
Promotes Traceability from Requirements to Capability to Functionality with Degree of Assurance
Satisfies
Security Capability
Mutually Reinforcing Security Controls
(Technical, Physical, Procedural Means)
Security Assurance
Developmental/Operational Actions
(Assurance-Related Controls)
Provides Confidence In
Produces
Generates
Security Functionality
Features, Functions, Services, Mechanisms, Processes, Procedures
(Functionality-Related Controls)
Security Evidence
Development Artifacts, Flaw Reports, Assessment Results, Scan Results, Integrity Checks, Configuration Settings

ISO/IEC 15026-2 Assurance Case

OMG Structured Assurance Case Metamodel (SACM)



IDA

INSTITUTE FOR DEFENSE ANALYSES

A Sample Security Assurance Case Pattern

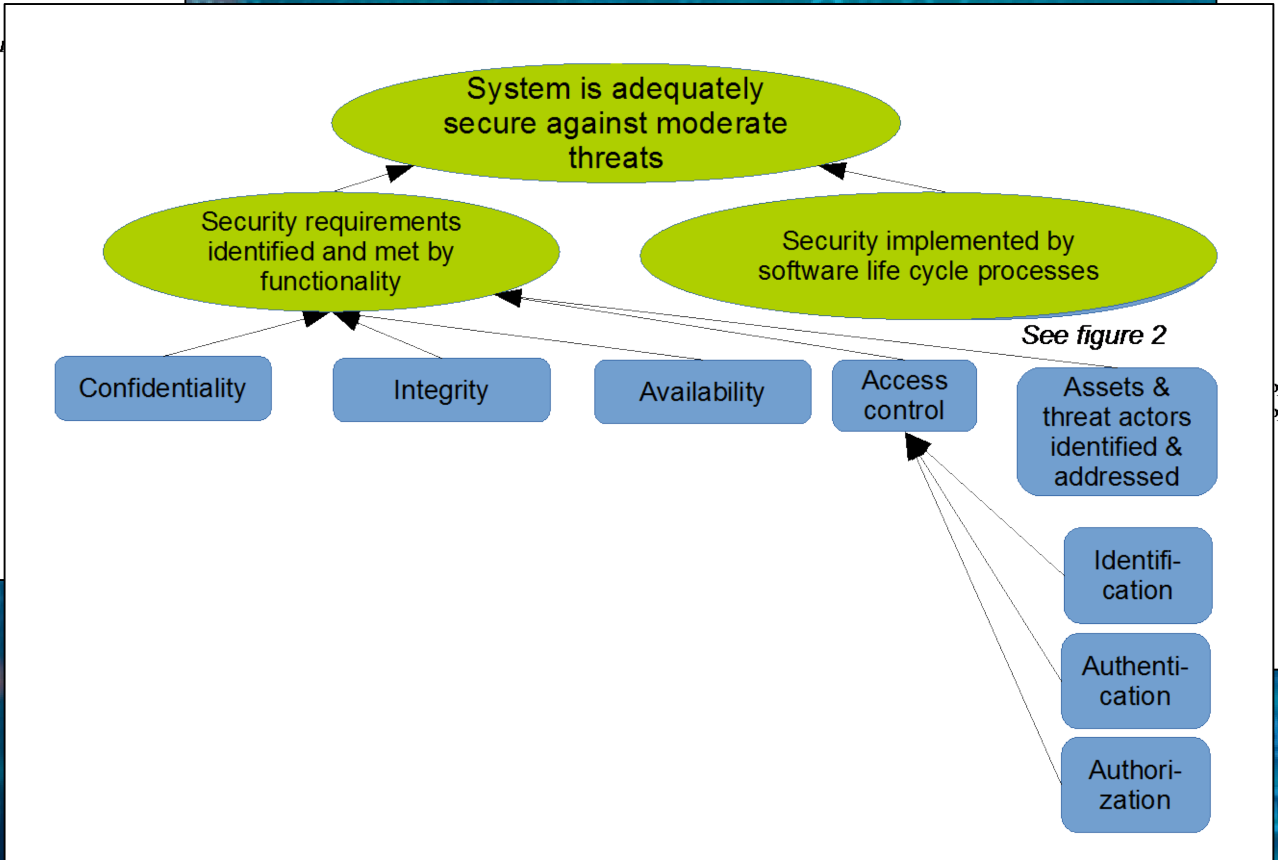
E. Kenneth Hong Fong, *Project I*
David A. Wheeler

December 2018

Approved for public release; distribution is unlimited.

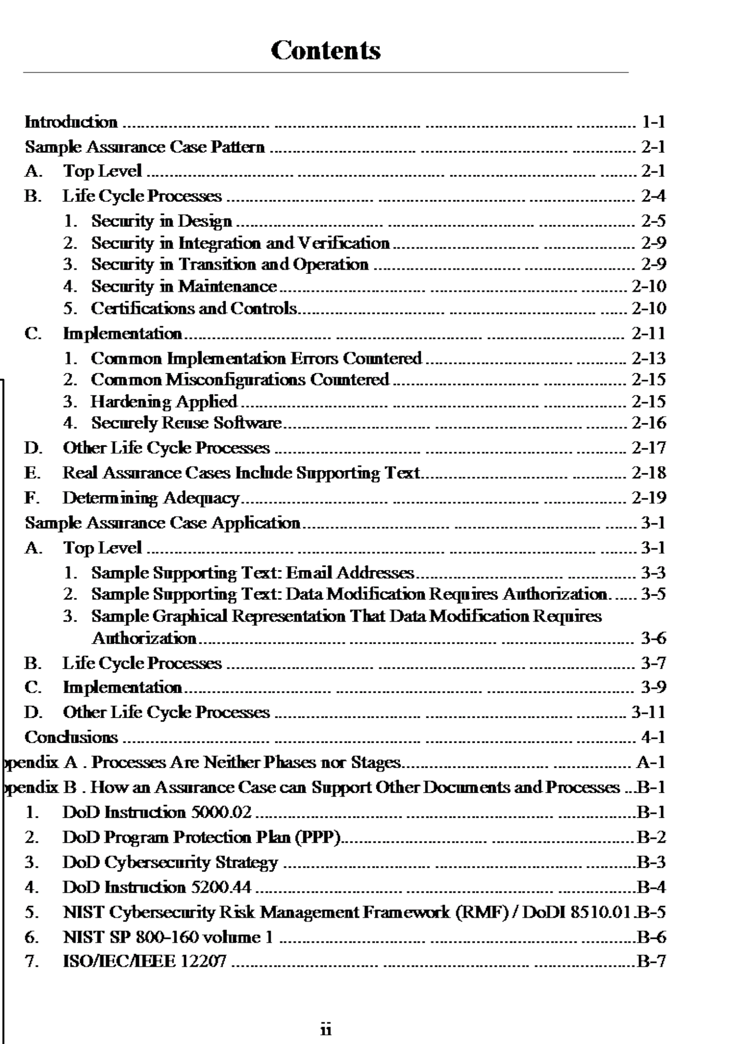
IDA Paper
P-9278

INSTITUTE FOR DEFENSE ANALYSES
4850 Mark Center Drive
Alexandria, Virginia 22311-1882



System is adequately secure against moderate threats
Security requirements identified and met by functionality
Security implemented by software life cycle processes
See figure 2
Confidentiality
Integrity
Availability
Access control
Assets & threat actors identified & addressed
Identifi-cation
Authenti-cation
Authori-zation

Figure 1. Top Level of an Assurance Case

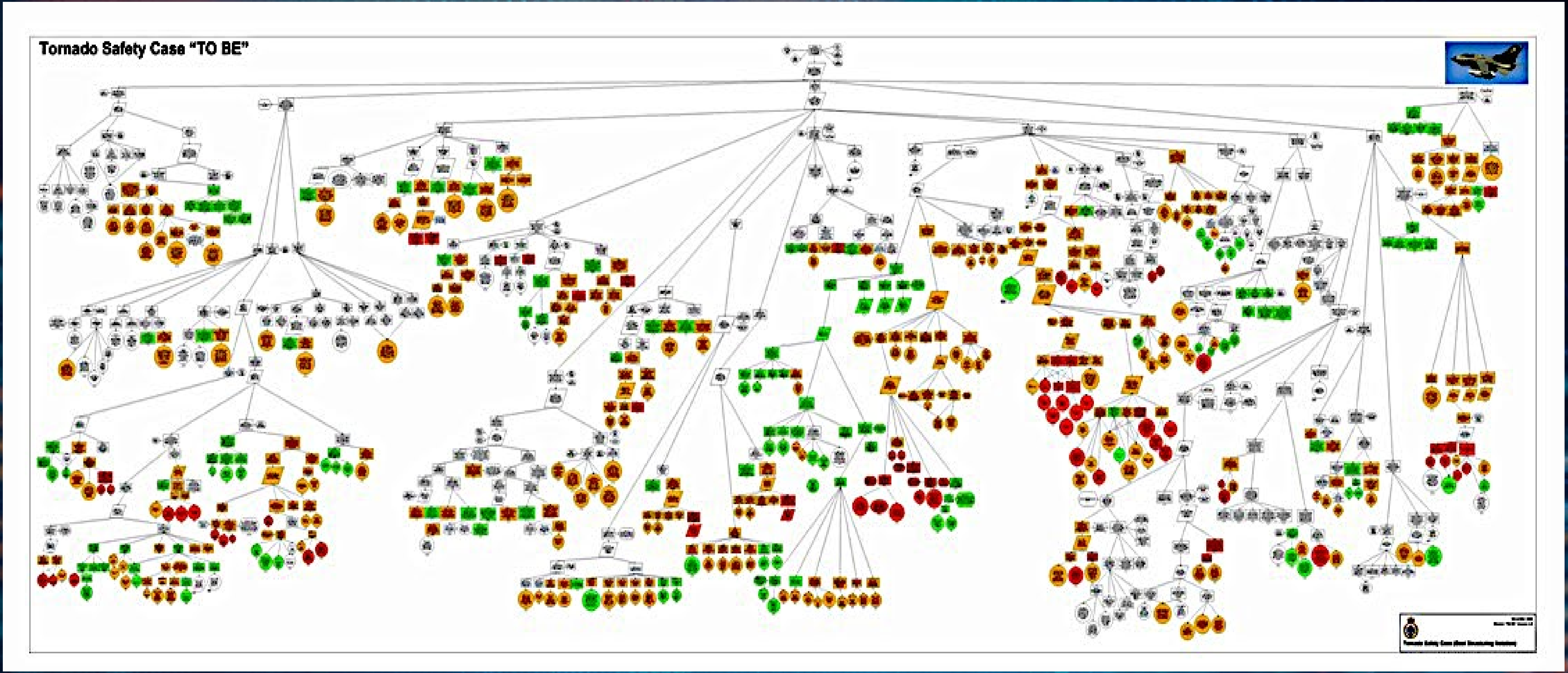


Contents

ii

https://www.ida.org/-/media/feature/publications/a/as/a-sample-security-assurance-case-pattern/p-9278.ashx

Tornado Operational Safety Case

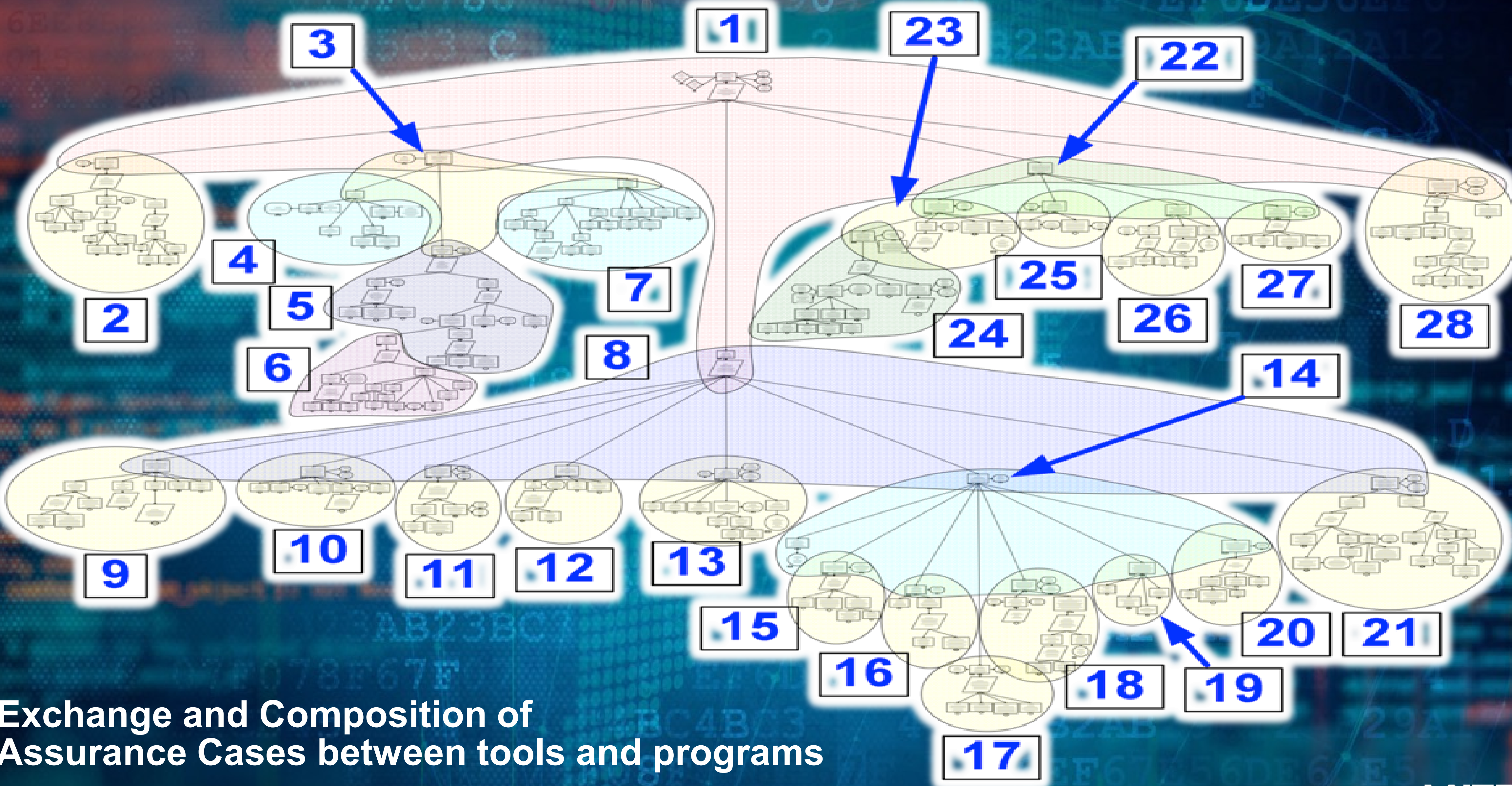


Apportionment of Ownership:

White = Generic, Green = Air Command, Orange = DE&S, Red = Contractors

MITRE

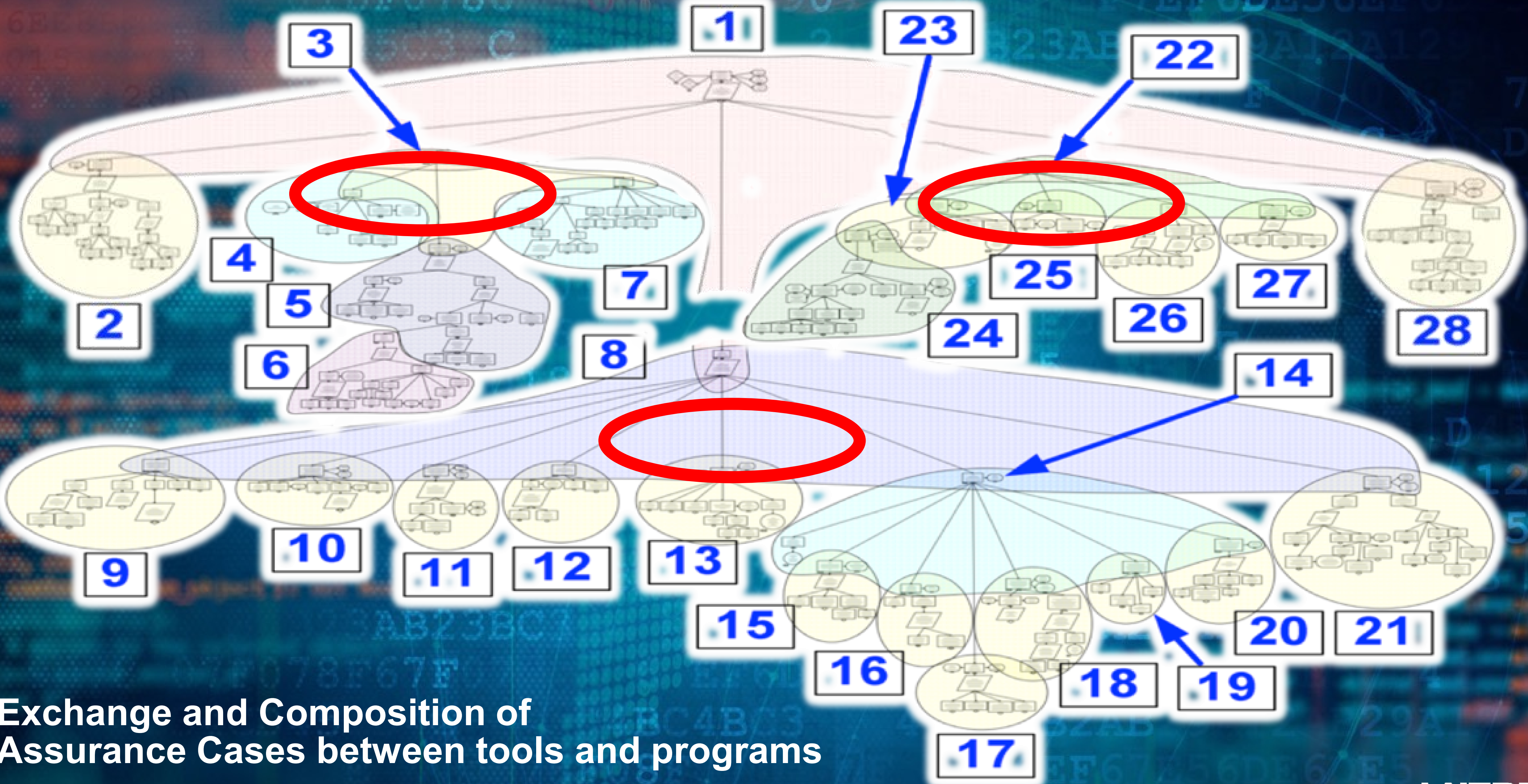
The Assurance Case for a System Builder using Assured Components



Exchange and Composition of
Assurance Cases between tools and programs

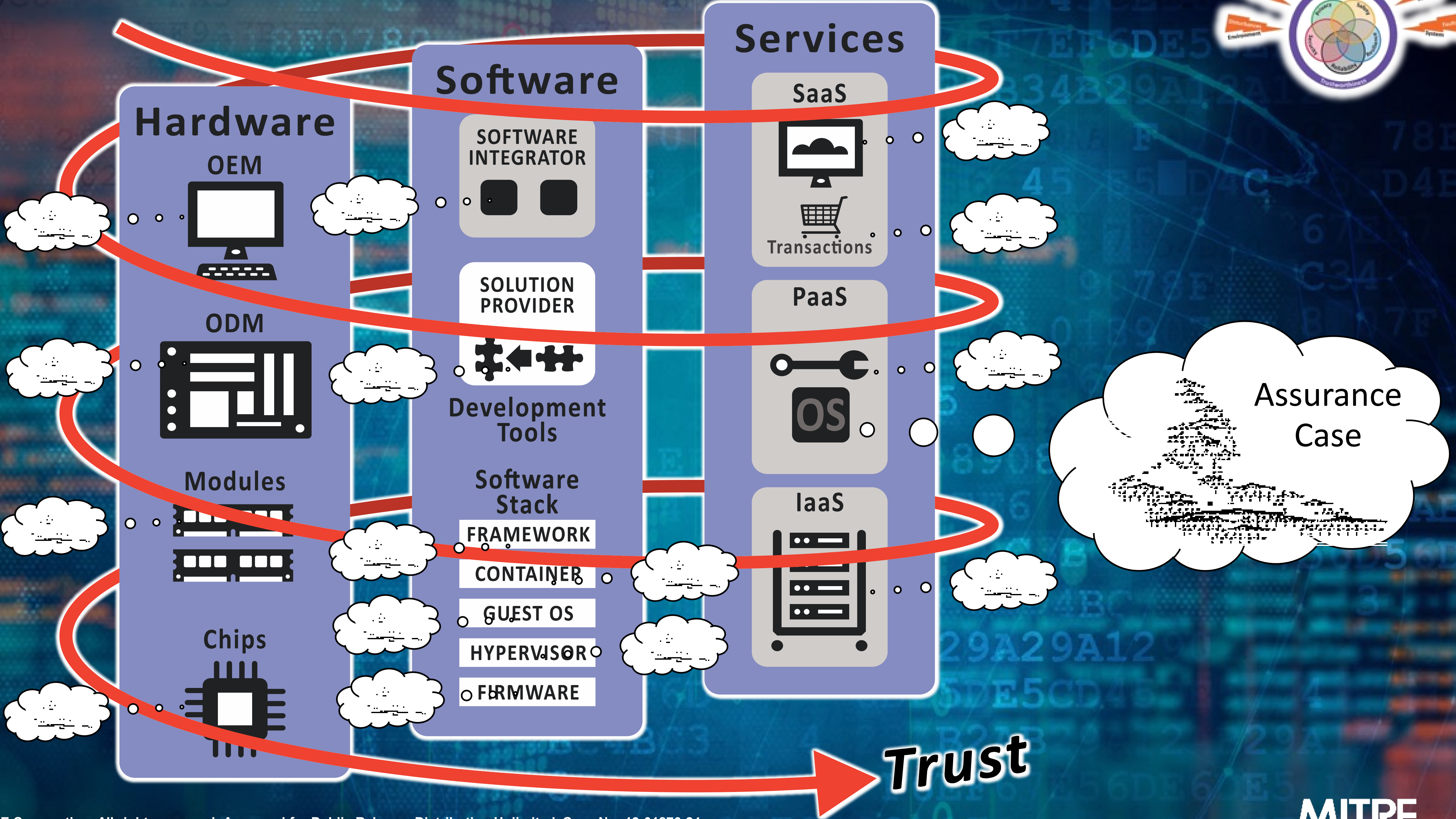
MITRE

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



MITRE

Questions?

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”

https://www.iiconsortium.org/news/joi-articles/2018-Sept-JoI_Assuring_Trustworthiness-FINAL2.pdf

industrial internet CONSORTIUM

www.iiconsortium.org

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

Authors:

Robert A. Martin
Senior Principal Engineer
The MITRE Corporation
ramartin@mitre.org

Ken Modeste
Director
Connected Technologies, UL LLC
Ken.Modeste@ul.com

September 2018 - 44 -

MITRE