Jeremy Bellay, PhD
Principal Data Scientist
Cyber Trust and Analytics
Battelle Memorial Institute

March 2, 2022

Composable Security

The Challenge of Security Models That Can Span from the Silicon to Software & Systems

BATTELLE

Context is critical in the realistic assessment of threats and mitigations

Threat

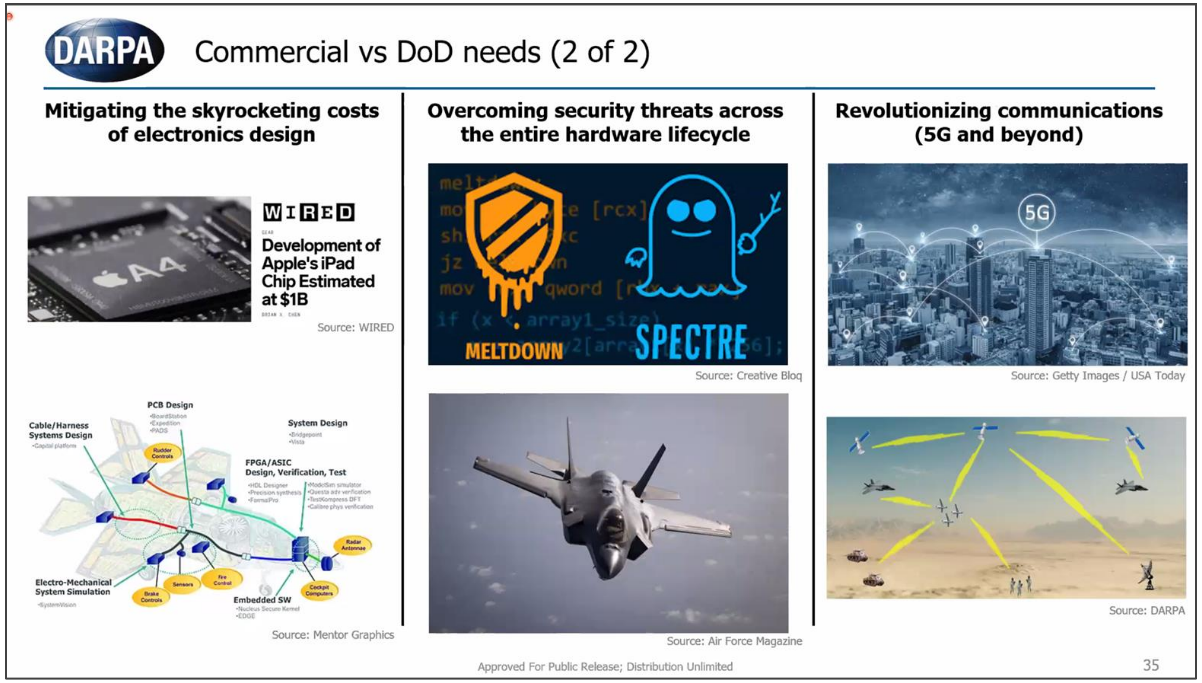
Existing Mitigation

A contextually ineffectual defense

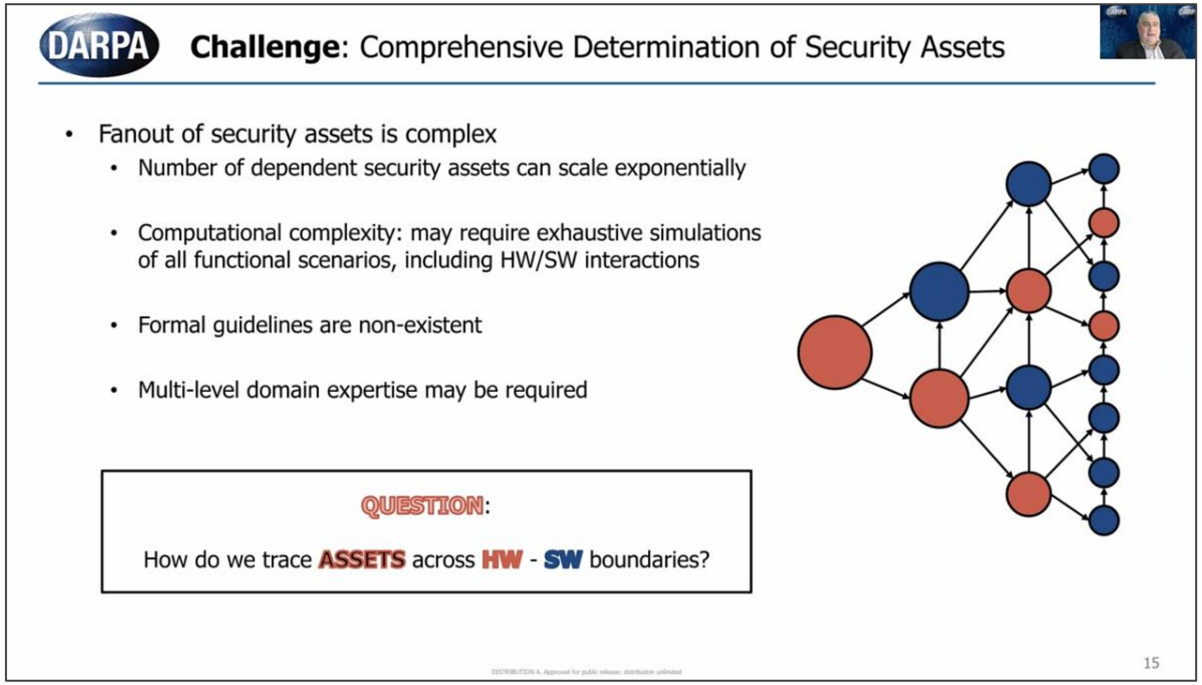
The system (or household) is necessary for estimating risk

BATTELLE

We know we need to understand security models at the system level



DARPA PM Carl McCants, NDIA Winter Meeting



DARPA PM Serge Leef, PAINE 2021

BATTELLE

Overview

- The problem of hardware and supply chain vulnerabilities
- Resources for security description and reporting
- An example vulnerability and mitigation across the lifecycle
- What do we need for system security?
- New and exciting BOMs
- Modular vulnerabilities
- Extensible assurance cases

TAME Hardware Assured and Weakness Collaboration and Sharing (HAWCS) Forum

TAME Forum Founder: Mark Tehranipoor (UF)
Lead: Jeremy Bellay (Battelle)
Co-Lead: Domenic Forte (UF)
Advisor: Bob Martin (MITRE)
Adviser :Jon Boyens (NIST)

Members:
Mike Borza Synopsys
Ron Perez Intel
Yatin Hoskote ARM
Sandip Ray University of Florida
Ioannis Savidis Drexel University
Fareena Saqib UNC Charlotte
Jon Graf Graf Research
Lisa McIlrath University of Florida
Mark Temmen US Army
Khalil Maloof ECI

Other TAME Working Groups:

Security and Trust Grand Challenges
Mark Tehranipoor (FICS)

Design for Security
Michel Kinsy (BU), Adam Kimura (Battelle)

THANK YOU!

Bellay, Forte, Martin, Taylor (2021) “Hardware Vulnerability Description, Sharing and Reporting: Challenges and Opportunities”. GOMAC

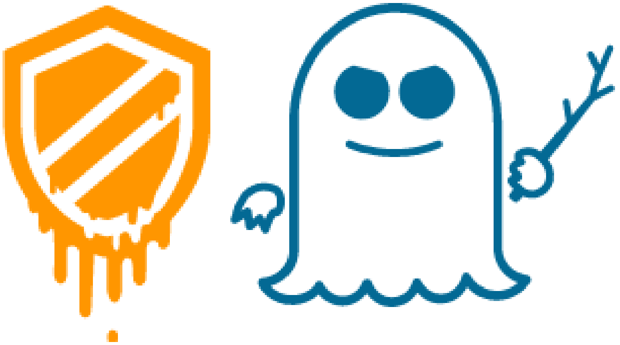
BATTELLE

Vulnerability Description Resources

“general-purpose hardware is fallible, in a very widespread manner, and this causes real security problems”

- Kim et al. 2014 (Rowhammer)

Hardware Vulnerabilities (Really Weaknesses)



Logical Weaknesses

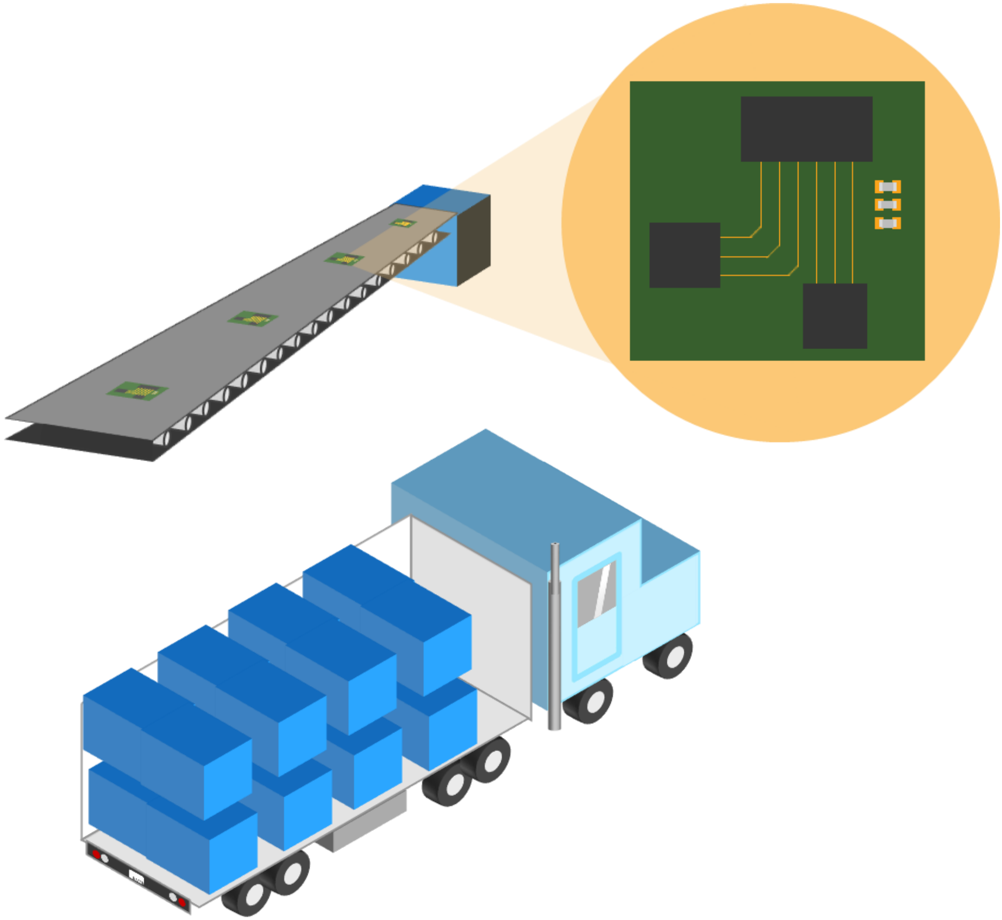
- Incorrect Register Defaults
- Internal Debug Mode Allows Override of Locks
- Spectre/Meltdown (2017) (Leakage from speculative execution)
- Starbleed (2020) (Decryption of bitstream)

Physical Weaknesses

- Rowhammer (2014) (Software induced memory faults)
- Glitching/Fault Attack (E.g. optical fault injection, 1966)
- Side-Channel Analysis (E.g. Differential Power Analysis,1996)

Hardware Security Issues

- Complex Logical Weaknesses
- Susceptibility to Reverse Engineering
- Counterfeits
- Third Party IP Integration and verification
- System integration with little visibility into components
- Supply Chain Issues



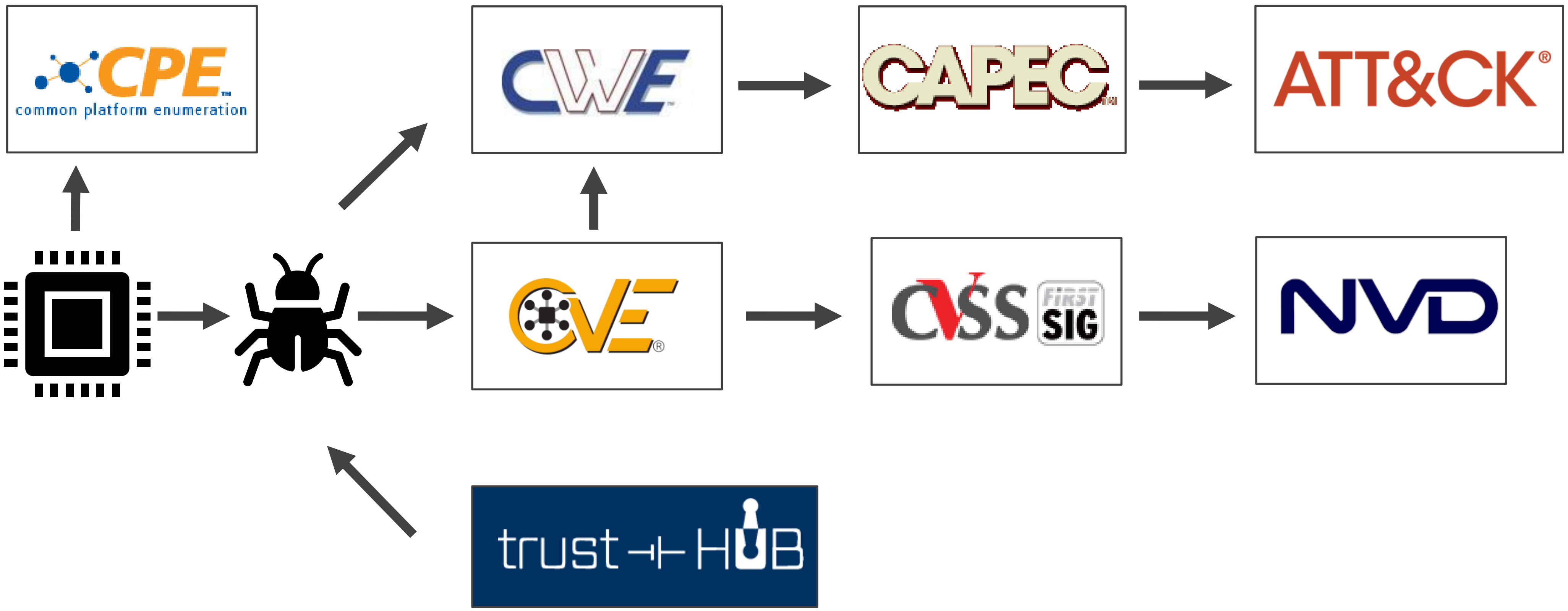
Hardware is hard to patch after manufacture, so vulnerability mitigation is pushed to the “left” of the lifecycle

BATTELLE

Current Security Description Frameworks

- CVE (Common Vulnerabilities and Exposures) - A naming space for identified vulnerabilities
- CWE (Common Weakness Enumeration). – A set of concepts and relations that describe the weaknesses that underly vulnerabilities
- CVSS (Common Vulnerability Scoring System) - A system for scoring the potential impact of a discovered vulnerability
- CAPEC (Common Attack Pattern Enumeration and Classification) A description framework for attack patterns
- CPE (Common Platform Enumeration) A naming scheme for IT systems, software, and components
- ATT&CK (D3FEND) – Catalog of attacker behaviors and mitigations, particularly in the case of APTs
- Trust-Hub - Contains a multitude of information supporting hardware assurance from taxonomies to exemplar data

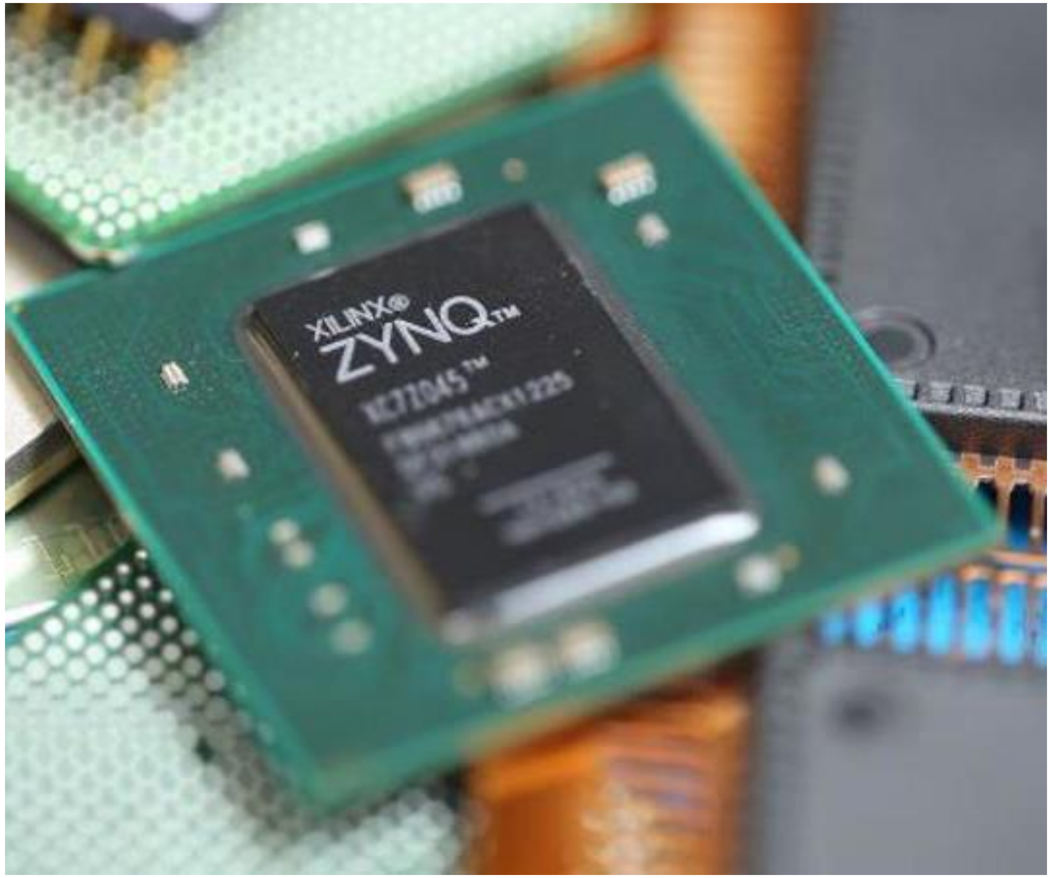
Vulnerability and Weakness Monitoring Description Resources



BATTELLE

Example: Xilinx SOC Zynq UltraScale+

- UltraScale+ Encrypt Only secure boot mode does not encrypt boot image metadata
- Disclosed 8/12/2019 (Xilinx issue 72588)
- Requires a ROM revision and is unpatchable
- Attackers can only exploit security flaw with physical access to a device, in order to perform a DPA (Differential Power Analysis) attack on the SoCs boot up sequence



How could this be reported using the current infrastructure?

CVE – CVE-2019-5478 - assigned by discoverer three weeks after disclosure

CPE - cpe:2.3:h:xilinx:zynq_ultrascale\+_mpsoc:-:*:*:*:*:*:*:*

CVSS – Base score reported - 5.5 (v3.1 MEDIUM), 2.1 (v2.0 LOW)

CWE-345 Insufficient Verification of data Authenticity

CWE-657 Violation of Secure Design Principles

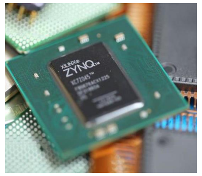
CWE - CWE-325: Missing Required Cryptographic Step

CWE - CWE-1300 Improper Protection Against Side Channel

CAPEC - CAPEC-242: Code Injection

CAPEC - CAPEC-622: Electromagnetic Side-Channel Attack (?)

Vulnerability and Weakness Monitoring Description Resources



cpe:2.3:h:xilinx:zynq_ultrascale\+_mpsoc:-:*:*:*:*:*:*:*

CWE-345 Insufficient Verification of data Authenticity

CWE-657 Violation of Secure Design Priciples

CWE-1300 Improper Protection Against Side Channel

CWE-325: Missing Required Cryptographic Step

Assigned CVE

CVSS (estimated 5.5)

CAPEC 242 Code Injection

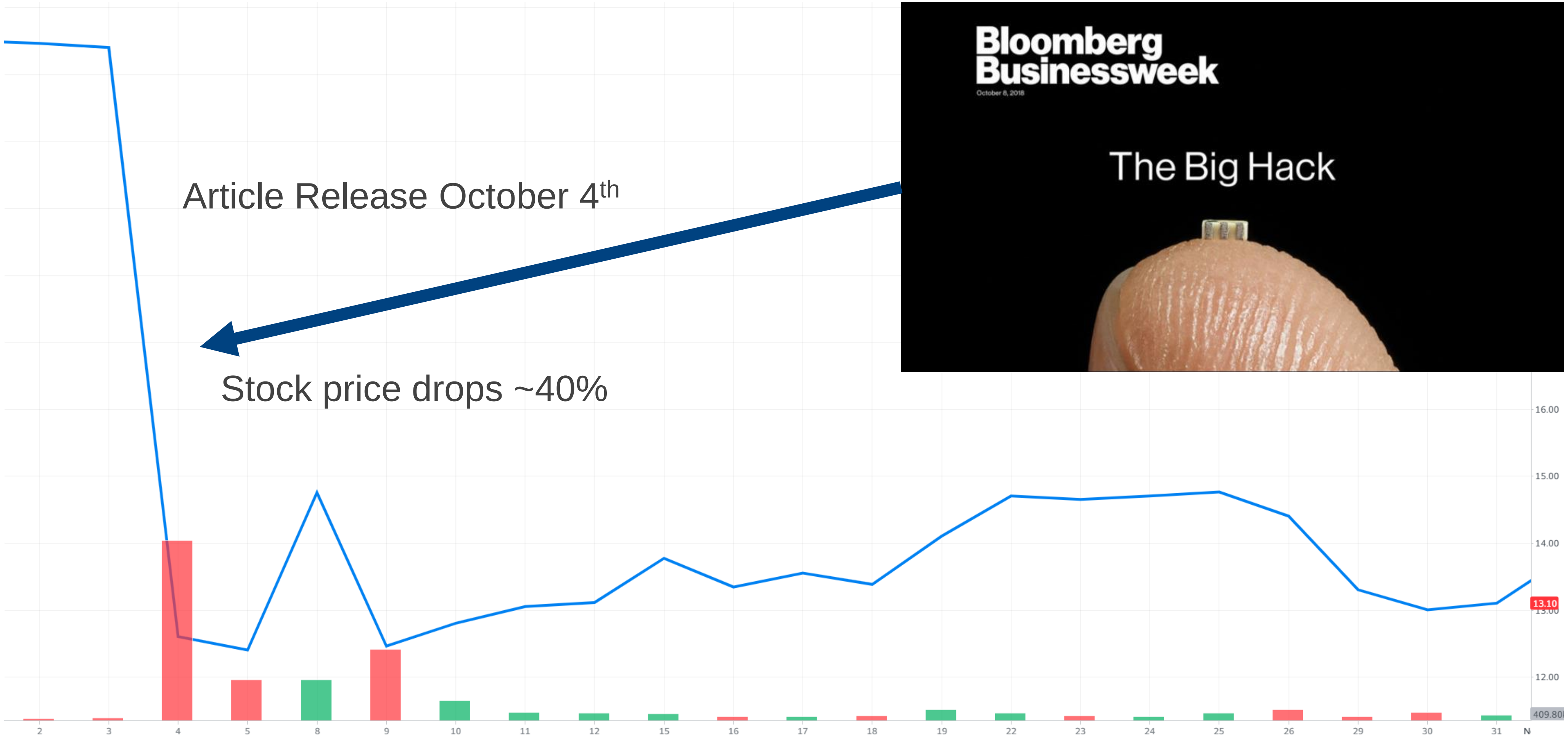
CAPEC 622 EM Side Channel

Vulnerability Disclosure

- Hardware vulnerability disclosure has been contentious due the unpatchability of many hardware vulnerabilities – not necessarily helped by new attention to hardware vulnerabilities
- Additionally, mitigation can be more complex because the inclusion of Original Equipment Manufacturers (OEMs) in addition to actual hardware vendor and relevant software distributors.
- IOT devices present several challenges for disclosure and mitigation
  - They are often inaccessible
  - Operate in non-standard environments
  - Small value/size may make sophisticated security feature undesirable
  - Only 10% of IOT vendors have a disclosure policy

BATTELLE

Vulnerability Release Without Responsible Disclosure Can Have Financial Consequences



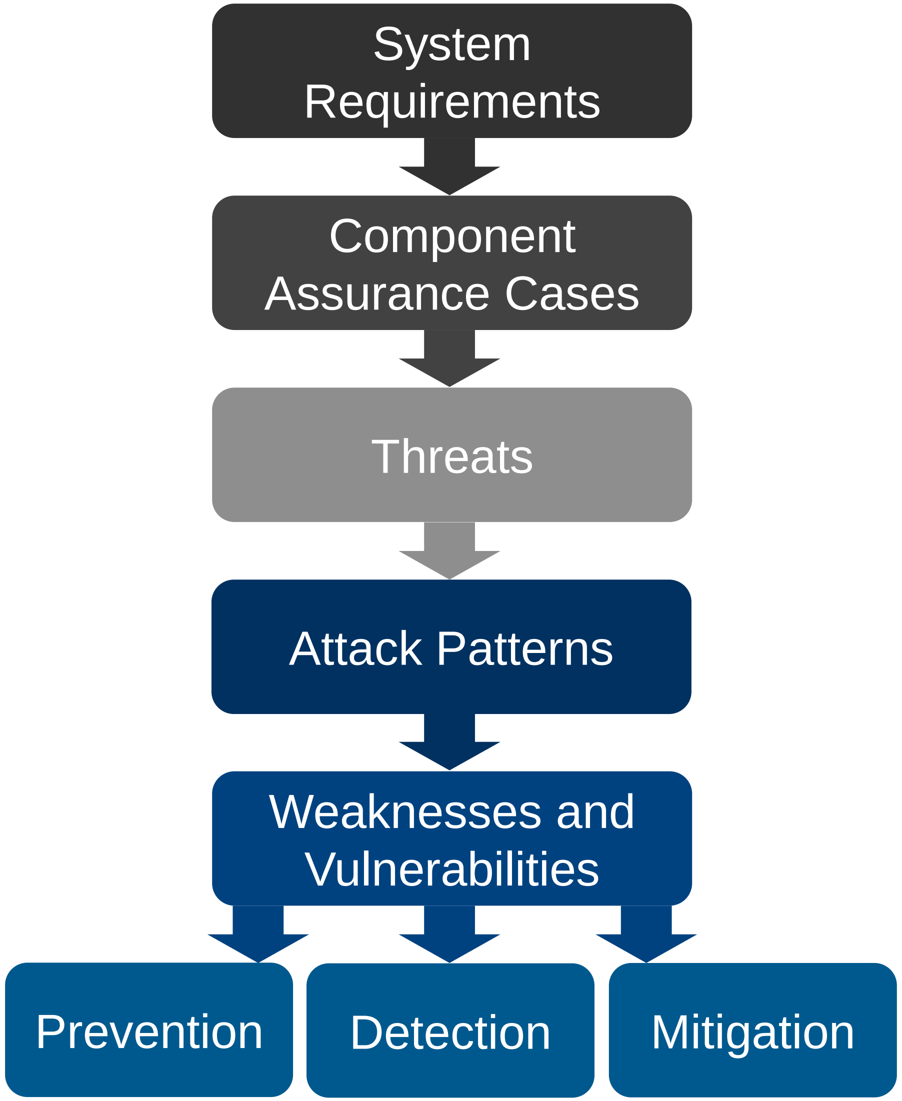
Article Release October 4th

Stock price drops ~40%

BATTELLE

Building the System Assurance Case

System Assurance Case



Protection of Sensitive Data

Data encrypted with 128bit encryption

Data Leakage
Key Leakage
Encryption disabled
} Data not encrypted

Physical Fault injection
Side Channel Analysis
Social Engineering
} Encryption disabled

Radiation Fault Injection (RFI)
Clock Glitch
Capacitive Coupling
} Fault injection

Redundant Key Operations
Formal Fault Detection
Shielding
} RFI Defenses

BATTELLE

System Assurance Case

Prevention	Removal of the manifestation of a weakness
Detection	Detect the presence weakness Detect the susceptibility of a weakness
Mitigation	Reduce the Severity of a weakness
Manifestation	When the underlying weakness presents itself
Exploitation	When an adversary utilizes the underlying weakness

BATTELLE

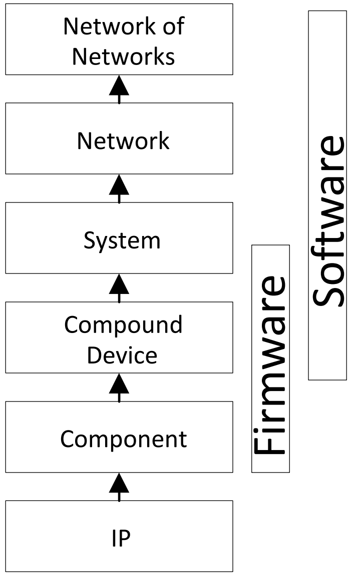
System Assurance Case

General Weakness	Manifestation	Exploitation	Mitigation	Detection	Prevention
Logical Design	X		X	X	X
Physical Design			X	X	
Hardware Device		X	X	X	
System Integration and Software Stack		X	X	X	

System Assurance Case

Fault injection	Manifestation	Exploitation	Mitigation	Detection	Prevention
Logical Design	✗		Redundant Key Operations	Formal Fault Detection	Generally, Not Completely Preventable
Physical Design	✗		Shielding		
Hardware Device		RFI Overpower Clock Glitch			
System Integration and Software Stack		Overpower Clock Glitch			

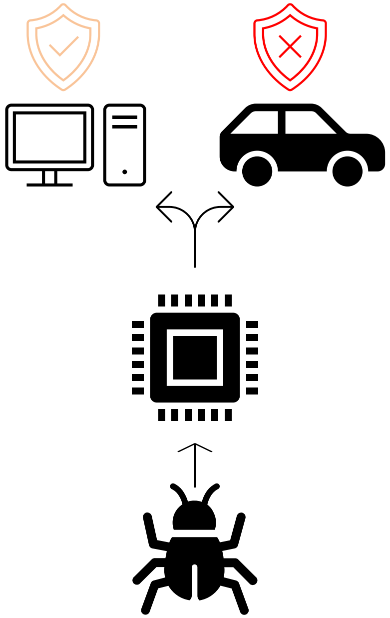
Pivot from description to integration



System Abstraction

Dependency in multiple directions: Security models must be applicable across the supply chain and up levels of system abstraction

Life Cycle Dependencies

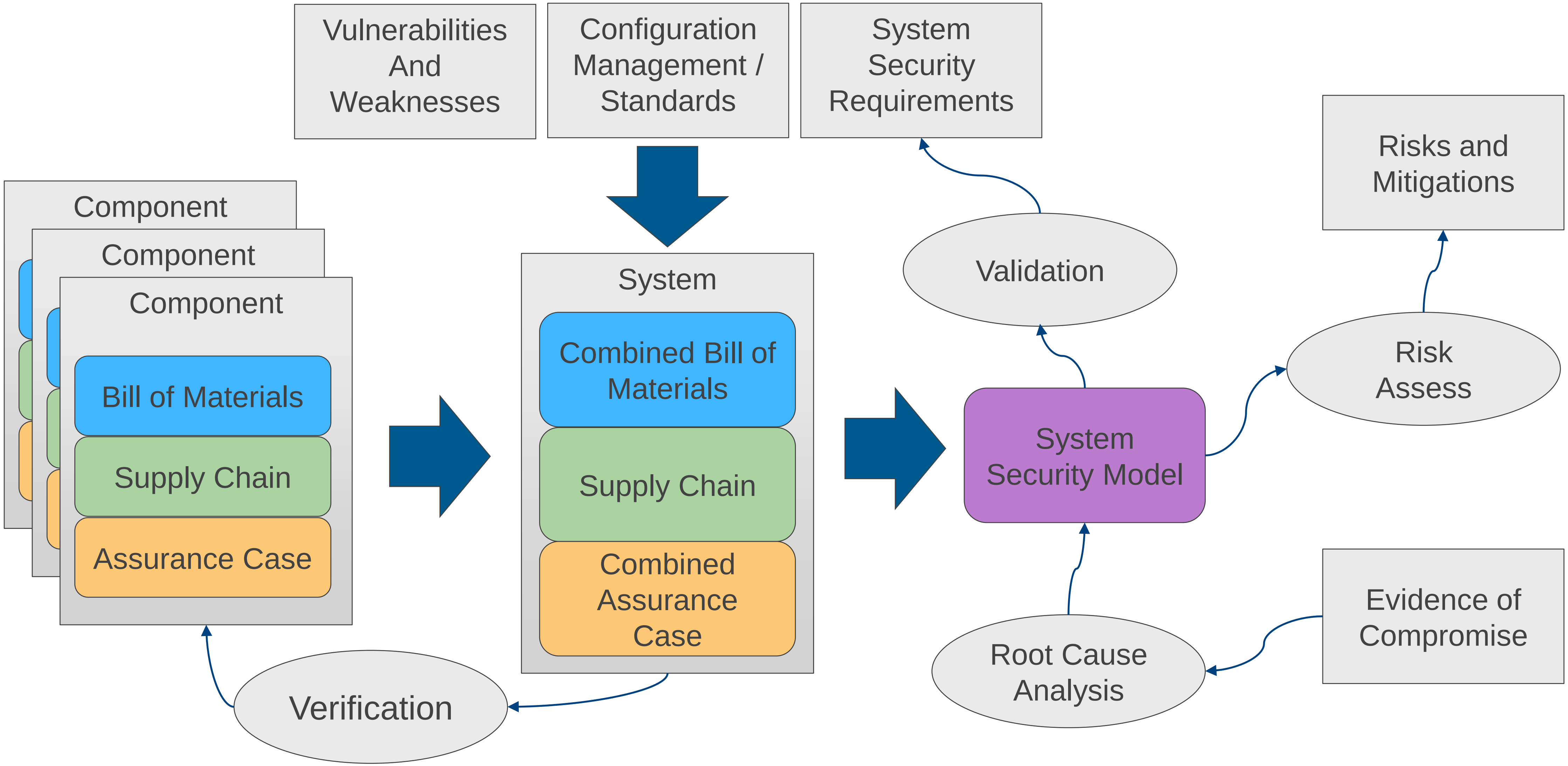


The context of component matters: A vulnerability that is only physically accessible might have high impact in certain contexts

Security models that are **composable** and **meaningful** across abstractions and context

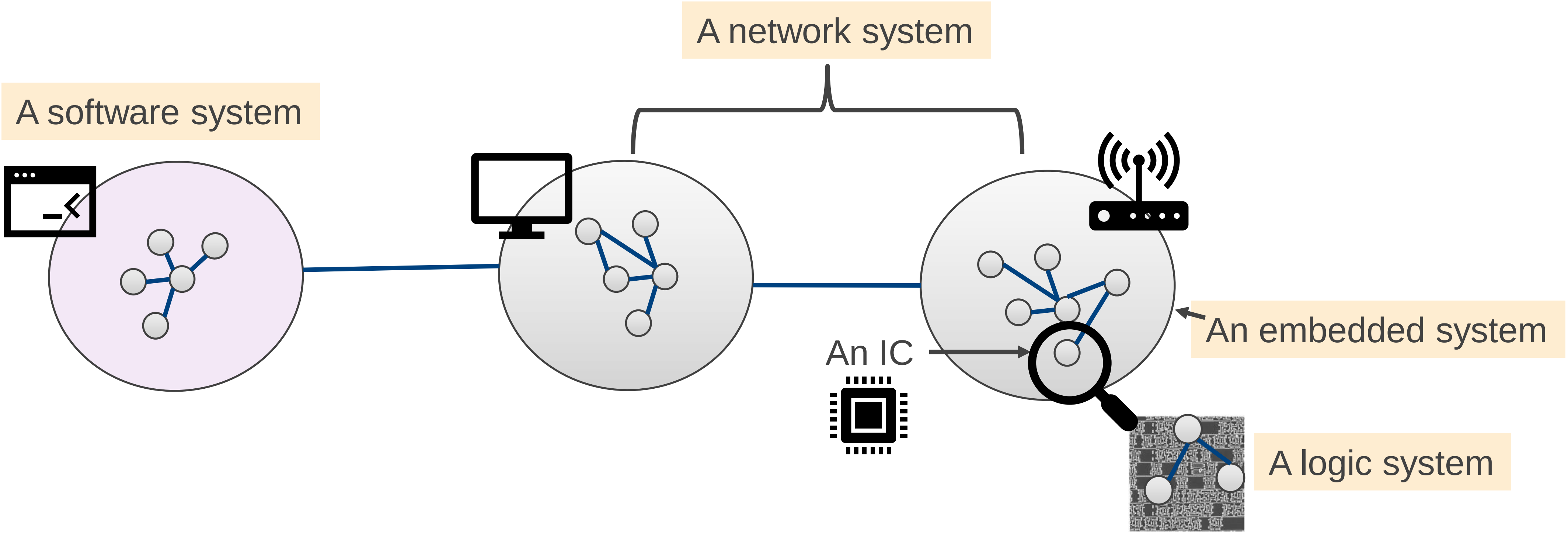
BATTELLE

From Component to System Assurance

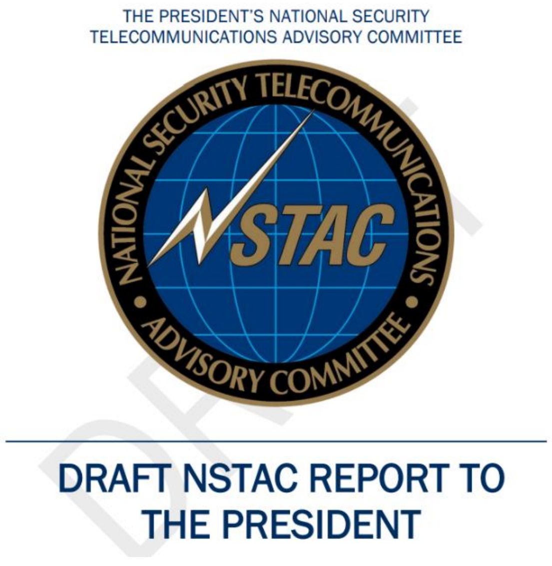


Systems of systems are intrinsic to cyber-physical threat modelling

Systems of systems contain hierarchical, lateral, and complex connections



The Software Bill of Materials Offers an Approach for Composite Systems



NTIA – “Minimum Elements For a Software Bill of Materials”

Data Formats: CycloneDX | SWID | PURL | CPE

Standards: SPDX | OpenChain | SEvA

Tools: In-toto | Dependency-Track | FOSSology | FOSSA | Grafeas

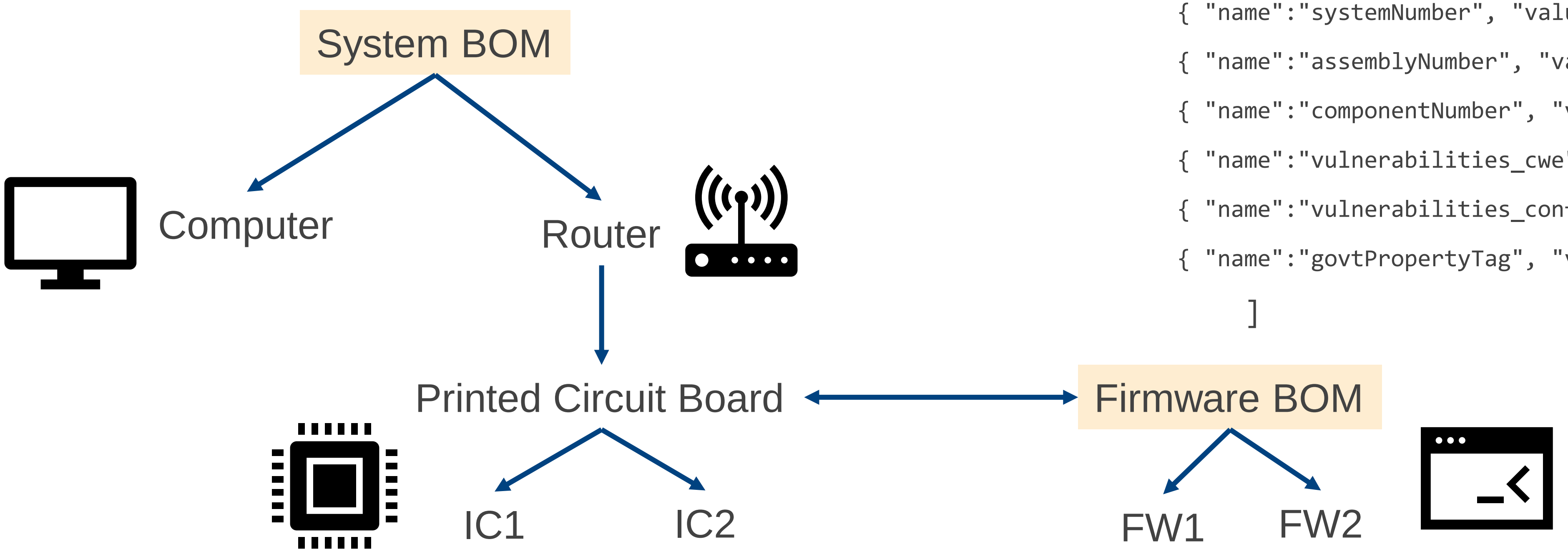
Could SBOM be extended to combined HW/SW systems?

BATTELLE

SBOMs can capture connections between system components

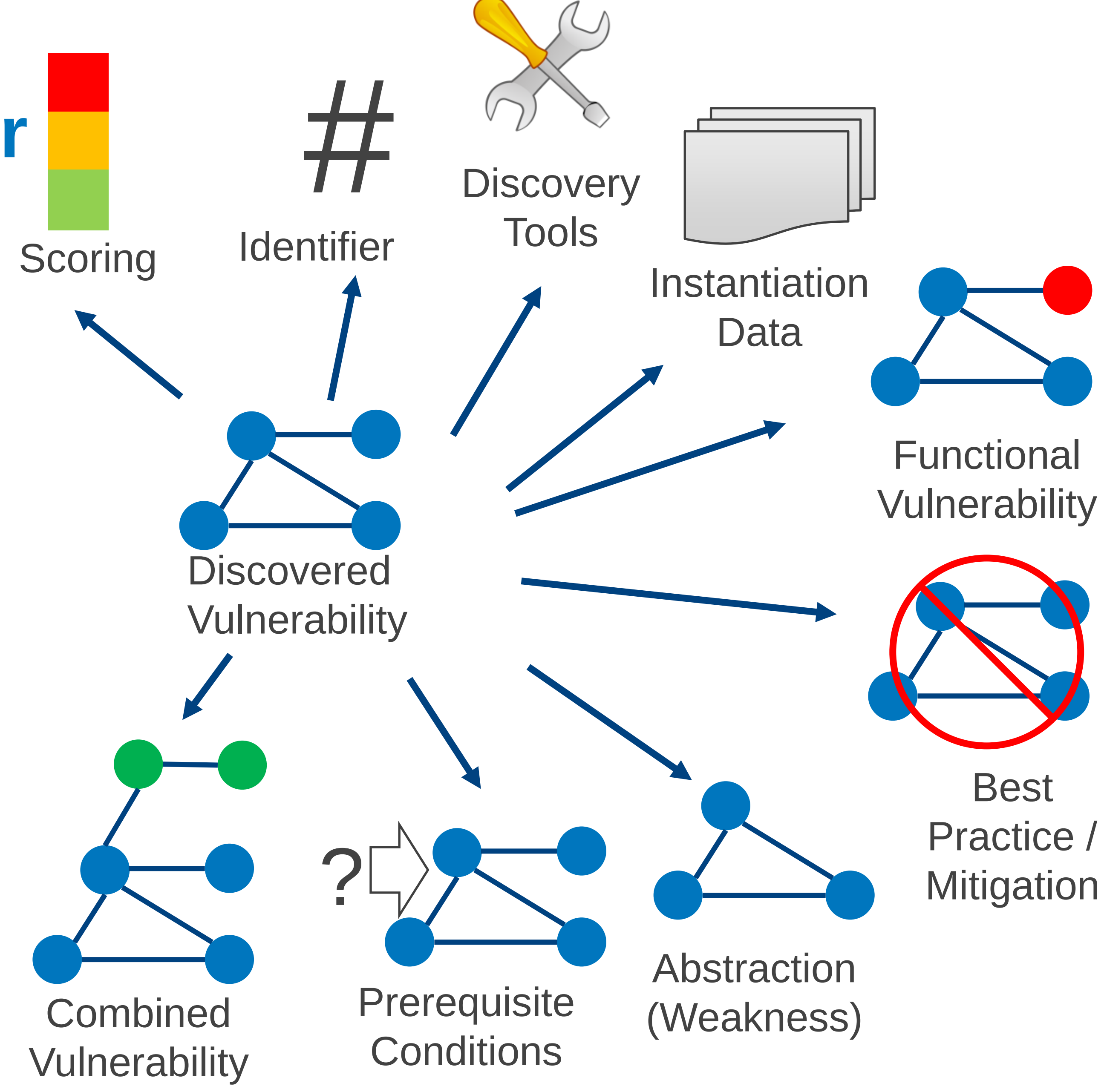
Example


```
"properties":[
    { "name":"bomType", "value": "2" },
    { "name":"systemNumber", "value": “XYZ0001" },
    { "name":"assemblyNumber", "value": "1" },
    { "name":"componentNumber", "value": “Z133" },
    { "name":"vulnerabilities_cwe", "value": "103,110" },
    { "name":"vulnerabilities_conf", "value": "0.88" },
    { "name":"govtPropertyTag", "value": "ABC123" }
  ]
```

What do we want from vulnerability description for system assurance?

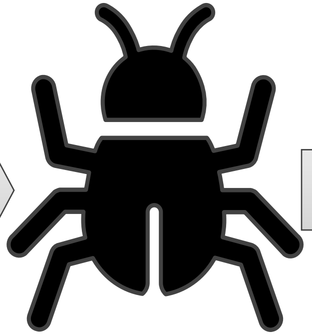
- Can we uniquely identify it?
- Can we describe it and abstract it?
- Where can it be found?
- What does it look like functionally?
- How can it be mitigated?
- How can it be prevented?
- How could it be accessed?
- How is it exploited?
- What is the consequence?



BATTELLE

What do we need to understand a vulnerability in a system?

Prerequisites
- Privileges Required
- Attack Vector
- Modality
- Capability Required
- User Interaction



Exploitation
- Complexity
- Indicators of Compromise
- Modality of Compromise

Consequence
- C,I,A Impacts
- Change of Privilege
- Change of Access
- Change of Modality
- Capability Change

CVSS Base Metric Group

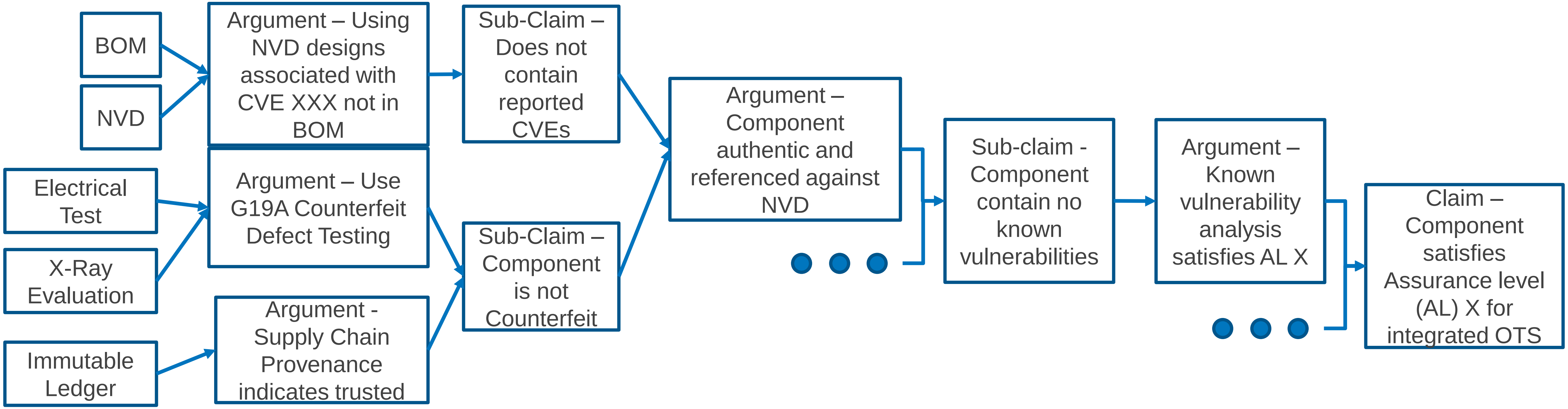
Exploitability Metrics
- Attack Vector
- Attack Complexity
- Privileges Required
- User Interaction

Impact Metrics
- Confidentiality Impact
- Integrity Impact
- Availability Impact
- Scope

BATTELLE

An assurance case that can span the lifecycle

- The Assurance Case formalism (ISO – 15026-2) is a data structure for representing claims, evidence, and arguments connecting the evidence to the claims.
- Assurance claims structured in this manner are **composable, reusable,** and can be **updated** gracefully
- The basis and results of a risk assessment can be encoded into the Assurance Case, as can the mitigation strategy and residual risk
- OMG Structured Assurance Case Metamodel (SACM) provides an XML structure for programmatic access



BATTELLE

Conclusions

- Descriptions of software and, more recently, hardware vulnerabilities have come a long way
- Vulnerabilities and mitigations interact throughout the lifecycle and levels of system abstraction
- To understand system security models, we need to describe what we have with an extended bill of materials
- Restructuring vulnerabilities would help to understand their system impact
- Assurance cases can extend throughout the lifecycle and system

bellayj@battelle.org

I would love to talk more in person at GOMAC 2022!

BATTELLE

BATTELLE

It can be done

800.201.2011 | solutions@battelle.org | www.battelle.org