CERT GBSD Projects: Designed in Assurance

Dr. Carol Woody, CERT Lead

October 21, 2021

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM21-0938

FY20-FY21 CERT Focus Areas for GBSD

Measurement for Assurance

*Zero Trust

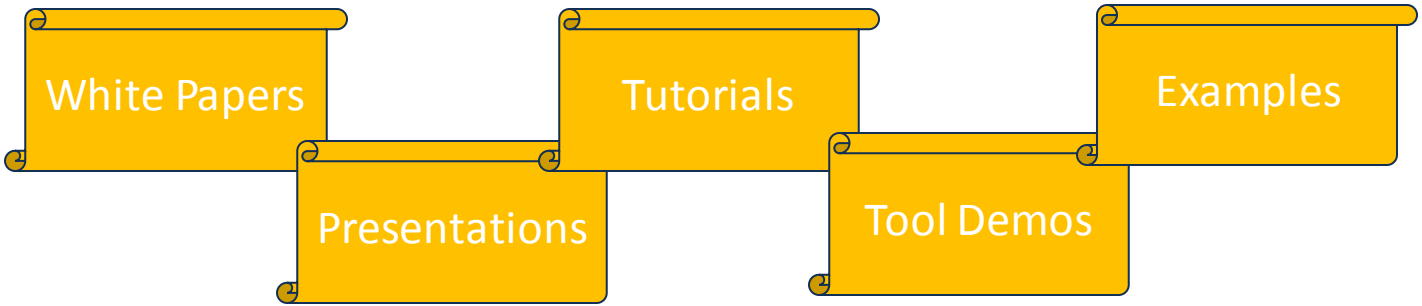
*Threat Modeling & Evaluation

*Applying Assurance Cases

Adoption of new research tools for DevSecOps

Acquisition Security Framework (ASF) Overview

Technical Debt for Cybersecurity



Selected for Today's Agenda

Integration of Cybersecurity into the Architecture for Designed in Assurance:

- Zero Trust: Tying it to Design [Tim Morrow]
- Linking Threat Modeling to Architecture Analysis (ATAM, QAW) [Natasha Shevchenko]
- Model-based Software Engineering for Cybersecurity: USAF High Assurance and DevSecOps [Carol Woody]
- STPA-SafeSec and Assurance Cases [John Goodenough]

Zero Trust: Tying it to Design

Timothy Morrow

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

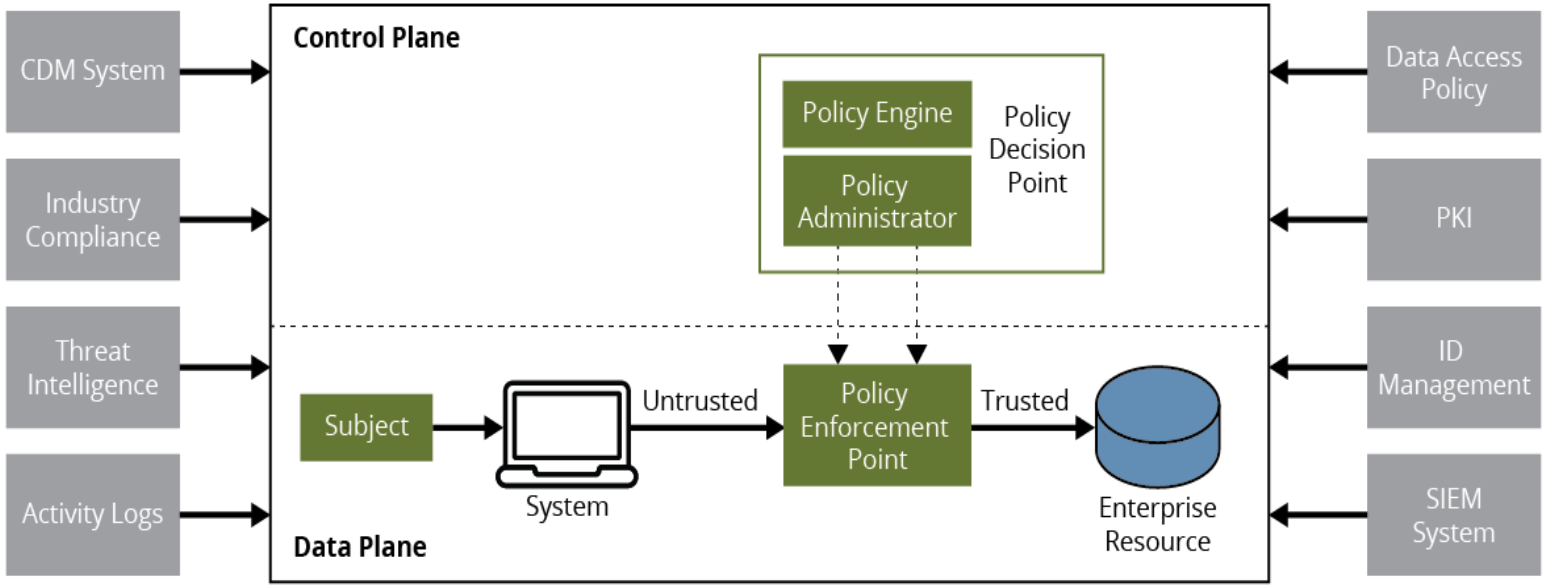
Zero Trust Tenets

Assume attacker presence.

Remove implicit trust in design and implementation.

Move security from the network to users, applications, and workloads.

Components (NIST SP 800-207)



Common Challenges

Governance

- Asset inventory

Architecture

- Awareness and accuracy

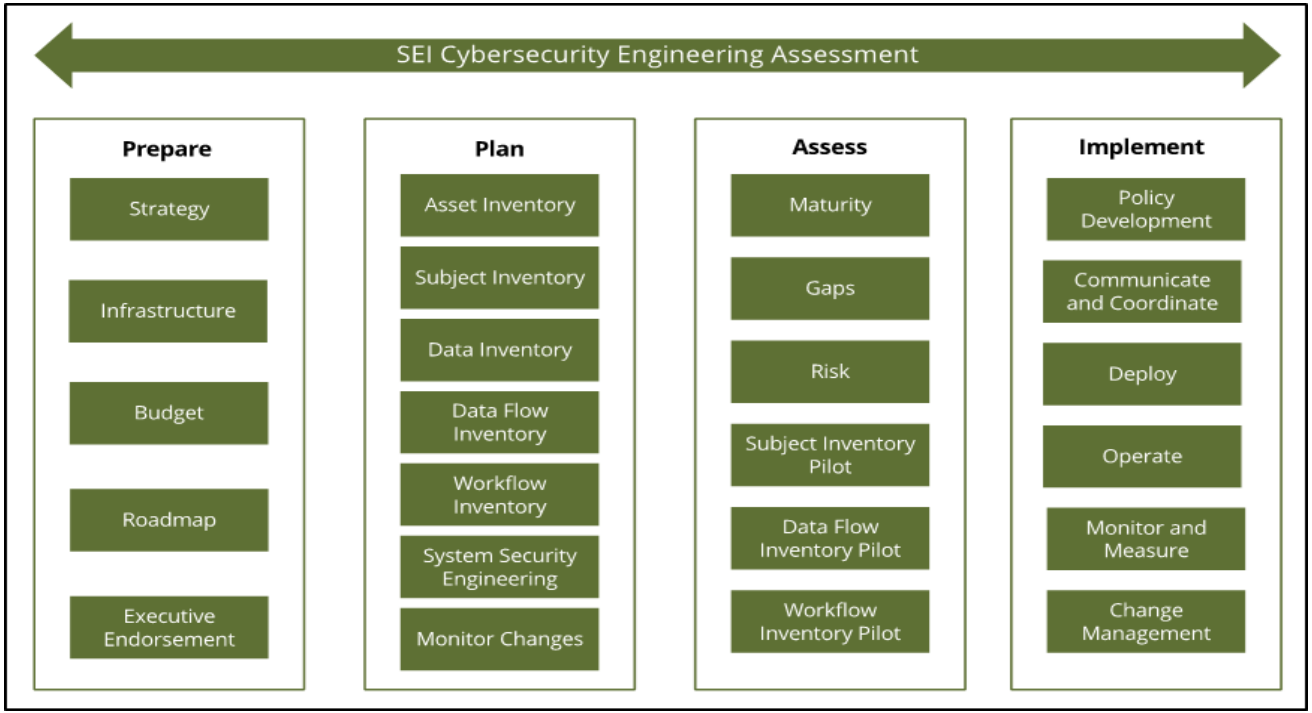
Cost

- Adoption cost

Measurement

- Success

Zero Trust Journey



Zero Trust Journey

SEI approach combines

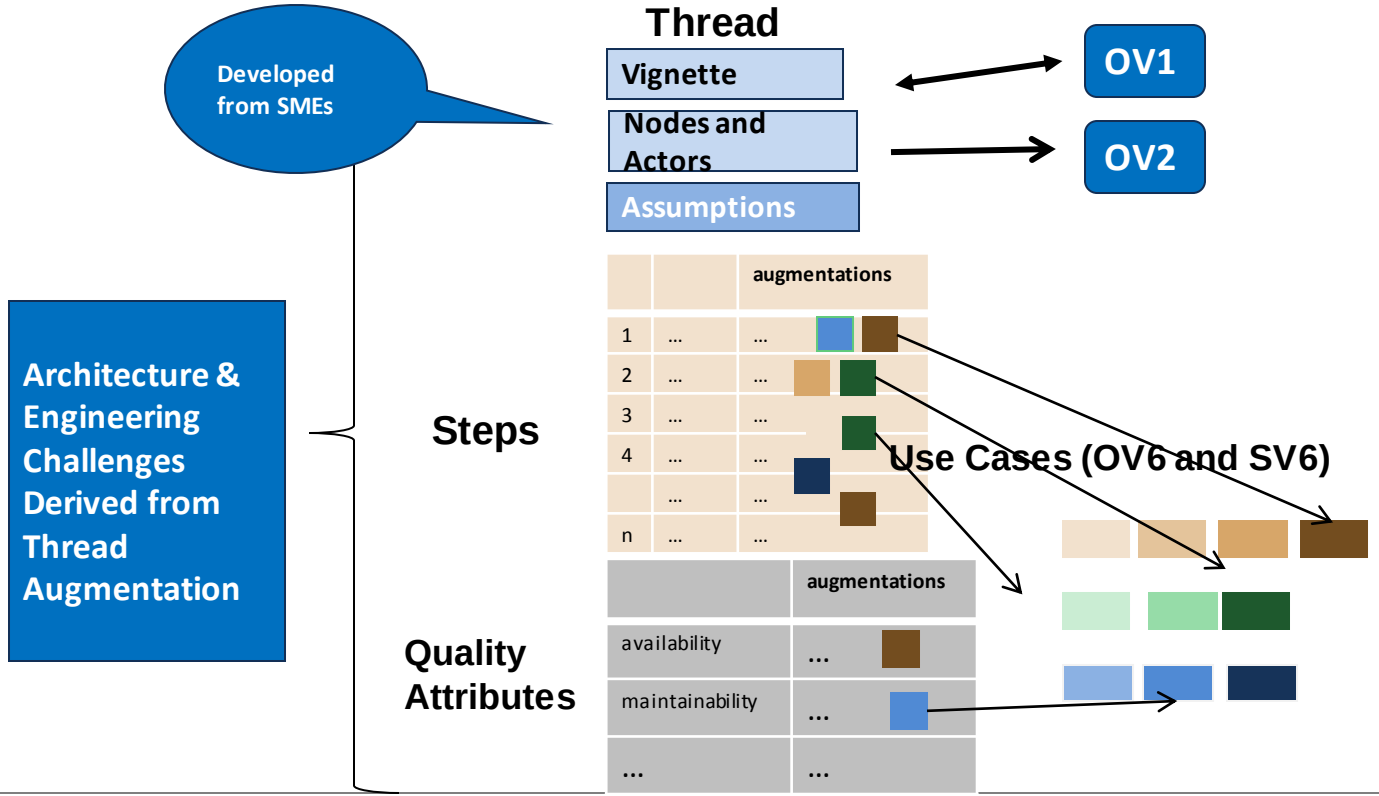
- Mission/Business Threads
- Systems Security Engineering (SSE)
- Model-Based Systems Engineering (MBSE)
- Continuous Authorization (cATO) concepts
- Cybersecurity Engineering Assessments

Mission/Business Threads

Artifacts that provide operational, lifecycle, and development context

- Vignettes
- Mission/business threads
- Architecture documentation

Mission Thread Template



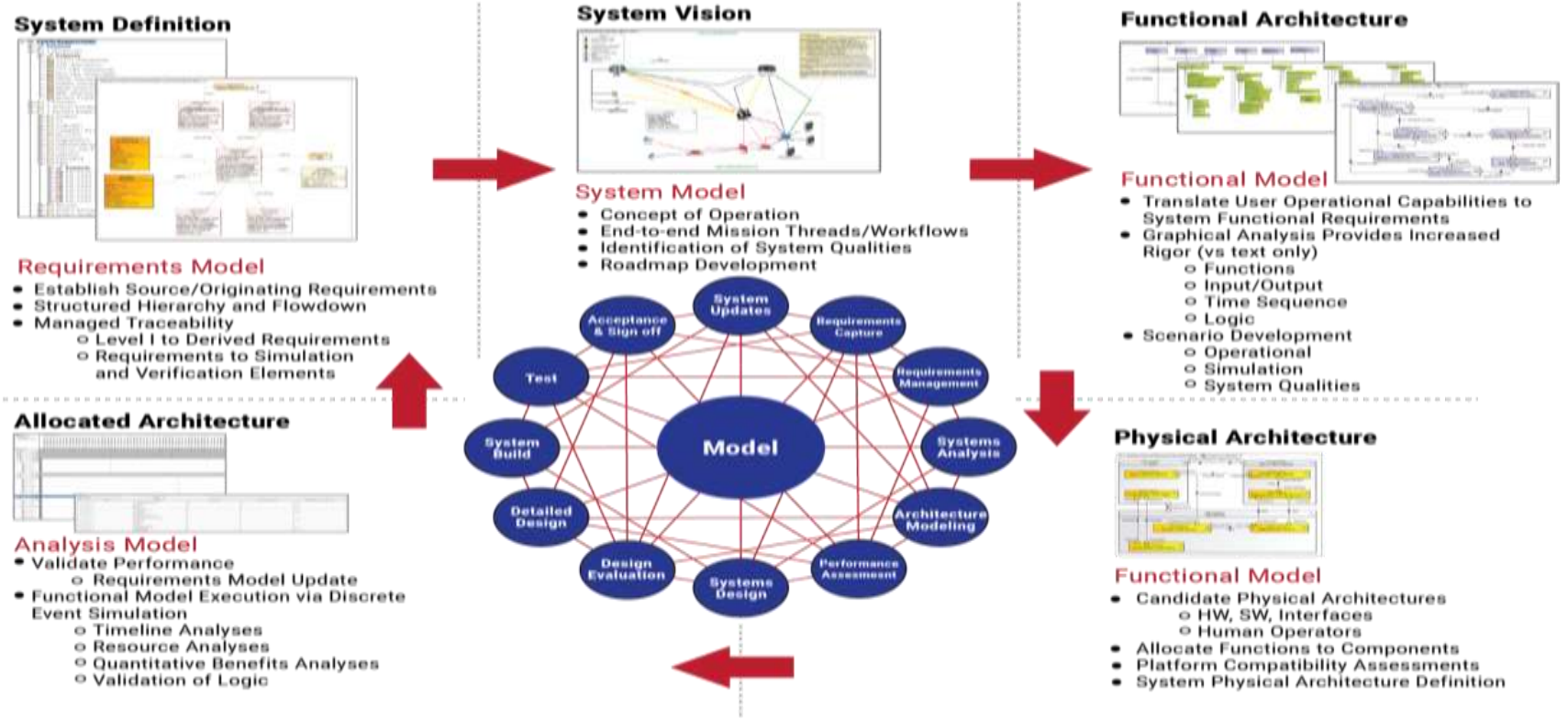
Systems Security Engineering

Process to achieve identified cybersecurity goals by building security in which supports analysis efforts.

Based on the following artifacts

- ISO/IEC/IEEE 15288:2015
- NIST Special Publication 800-160, Volume 1
- NIST Special Publication 800-160, Volume 2
- NIST Special Publication 800-37

Model Based Systems Engineering (MBSE)



Continuous Authorization to Operate (cATO)

Incorporates the NIST Risk Management Framework (RMF) and continuous monitoring with software engineering activities that leverage cloud computing and cyber-resilient systems engineering.

Key Conditions

1. Adoption and deliberate use of a secure software supply chain.
2. Complete understanding of activities inside system boundaries including robust continuous monitoring.
3. Ability to conduct active cyber defense in order to respond to cyber threats in real-time.

** CrossTalk August 2021, "Exploring the Ingredients of a Continuous Authorization to Operate", Weiss, J. and Gesling, T.*

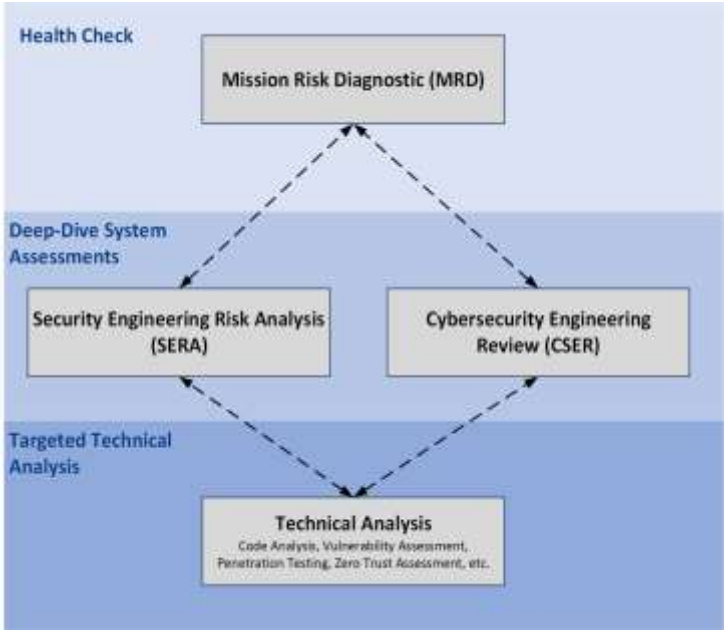
Cybersecurity Engineering Assessments

SEI is developing an integrated approach for assessing and managing security across the system lifecycle and supply chain.

Health check.

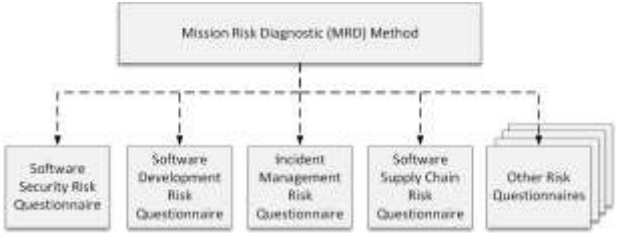
Deep-dive system assessments.

Targeted technical analysis.



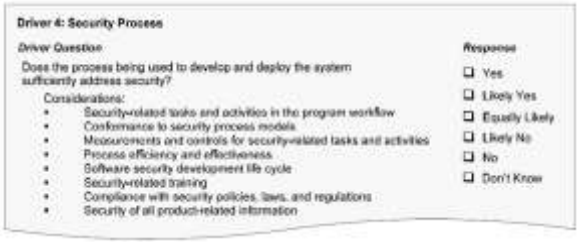
MRD Method

MRD Platform



Risk Factors

Risk Factor Evaluation



Driver 4: Security Process

Driver Question

Does the process being used to develop and deploy the system sufficiently address security?

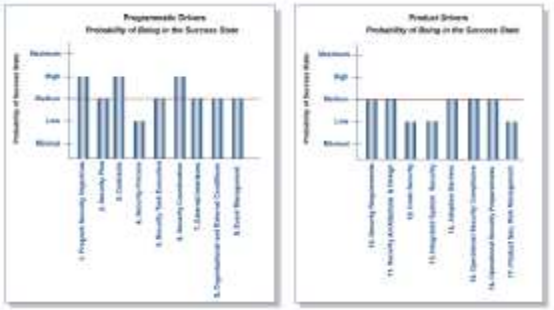
Considerations:

- Security-related tasks and activities in the program workflow
- Conformance to security process models
- Measurements and controls for security-related tasks and activities
- Process efficiency and effectiveness
- Software security development life cycle
- Security-related training
- Compliance with security policies, laws, and regulations
- Security of all product-related information

Response

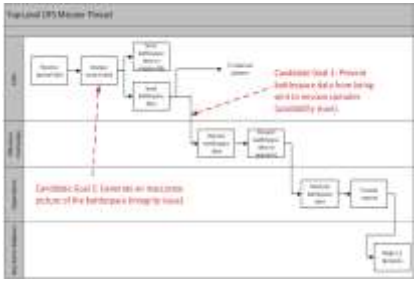
- ☐ Yes
- ☐ Likely Yes
- ☐ Equally Likely
- ☐ Likely No
- ☐ No
- ☐ Don't Know

Mission Assurance Profile

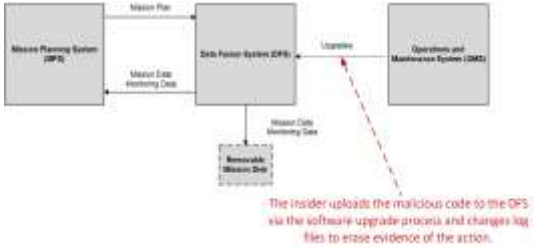


SERA Method: *Example*

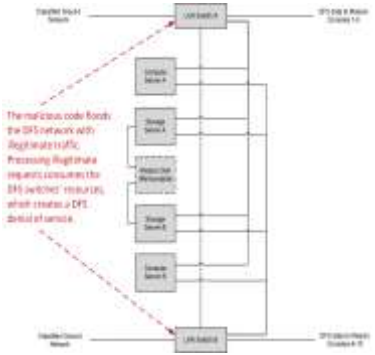
Mission Thread



System Interfaces



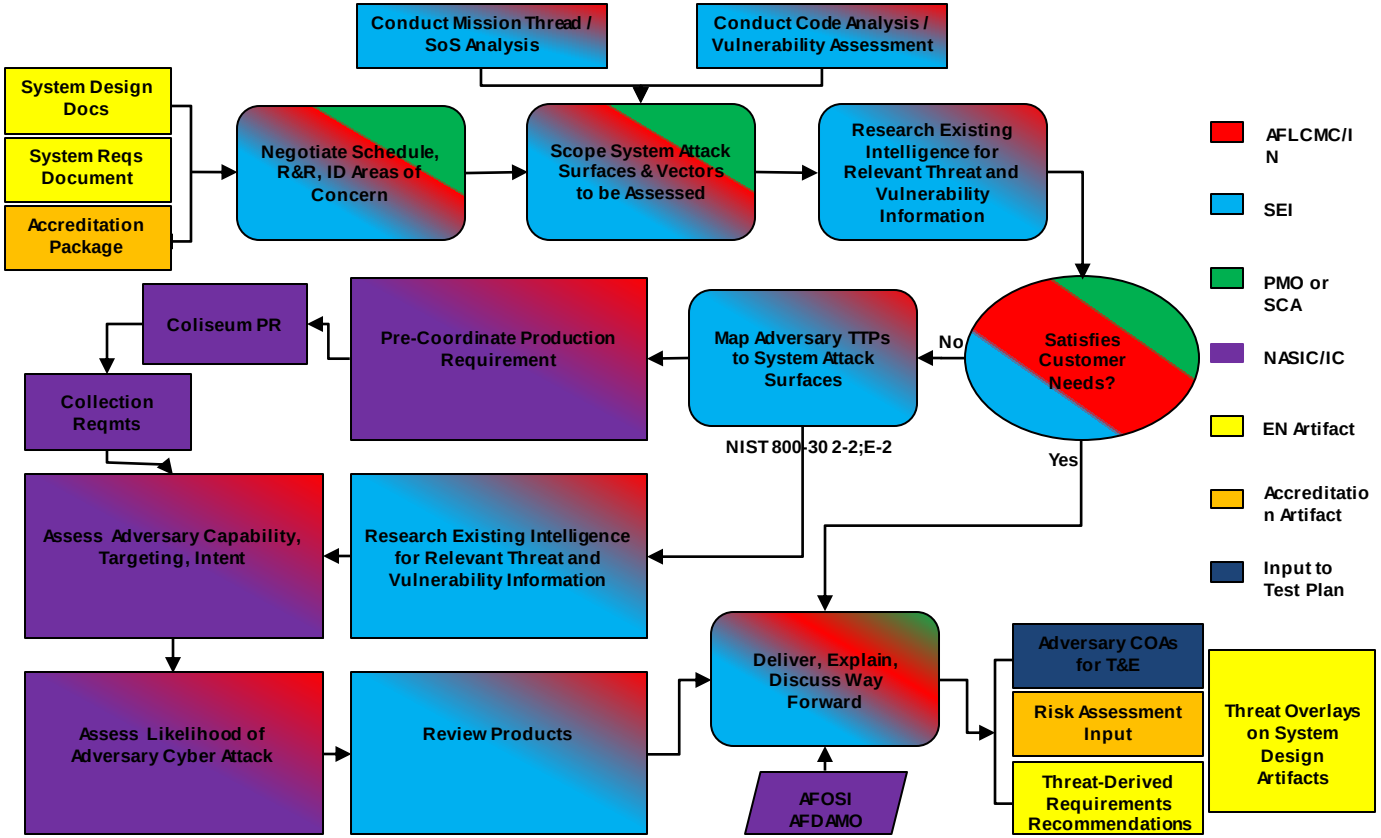
System Architecture



Threat Profile

Step	Enabler	Candidate Control	SWOT Reasoning
1.	Insufficient feedback about employee performance	The organization's managers are trained to provide meaningful feedback on performance issues.	
2.	Extension for inappropriate employee behavior	The organization's managers recognize inappropriate behavior when it occurs and respond appropriately.	
3.	Has intention to undertake employee issue	The organization's managers recognize an employee's escalation/retaliation and proactively work to diffuse the situation.	
4.	Insufficient access control for information and resources (physical and cyber)	Physical access to information and resources is managed appropriately.	
		Access permissions and authorization to computing resources are managed.	
	Insufficient monitoring of the organizational environment for abnormal activity (physical and cyber)	The organization monitors the physical environment for abnormal activity.	
		The organization monitors systems and networks for abnormal activity.	
		The organization performs targeted monitoring of individuals with suspected behavioral issues.	
		The organization responds appropriately when abnormal activity is detected.	

Hybrid Adversary Cyber Threat Assessment (ACTA) Process



Hybrid Adversary Cyber Threat Assessment (ACTA) Process

ACTA/SERA process is used to unify efforts of the engineering, acquisition intelligence, and intelligence production communities in delivering “decision quality” threat to inform system design and risk management decisions.

The process allows for traceability to NIST SP 800-30; designed to satisfy the DoD mandate to implement RMF (DoDI 5000.02, 8510.01) & RA (DoDI 8500.01)

1. ACTA begins with ingest of program artifacts to include design documents, requirements documents, and accreditation packages.
2. SEI/IN conducts Mission Thread / System-of-Systems analysis to supplement program artifact review.
3. IN/SEI works with customer to scope and prioritize system attack surfaces to be assessed.

Hybrid Adversary Cyber Threat Assessment (ACTA) Process

4. IN/SEI conducts organic research to identify existing cyber threat intel and align adversary tactics, techniques and procedures (TTPs) to system attack surfaces.
5. SEI/IN conducts code analysis and vulnerability assessment to identify exploitable vulnerabilities that align with adversary TTPs.
6. IN will use findings from preliminary research to develop a production requirement for the IC.
7. Results from the IC analysis are then used to assess adversary capability, targeting, and intent, and to assess the likelihood of threat initiation.
8. IN/SEI translates the results into threat overlays on program office artifacts such as DoDAF Operational Views, System Views, etc.
9. IN/SEI present tailored ACTA findings to Program Office, enabling future program decisions and risk mitigations

CSE Lifecycle Roadmap

A collection of cybersecurity engineering practices and competencies that can be applied across a system lifecycle.

1. Security risk assessment.
2. Requirements.
3. Architecture and design.
4. Implementation.
5. Developmental test and evaluation (DT&E).
6. Operational test and evaluation (OT&E).
7. Operations and sustainment (O&S).

Each area includes
- *Practices*
- *Evidence*
- *Competencies*

Zero Trust

Questions

Threat Modeling & Evaluation

Natasha Shevchenko

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Agenda

What is Threat Modeling?
Types of Attacks
Threat Modeling Method
Threat Modeling for SoS
Threat Modeling in SDLC
Threat Modeling in Agile
Threat Modeling in DevOps
Q&A

What is Threat Modeling? - 1

"Threat modeling is a process by which potential threats, such as structural vulnerabilities can be identified, enumerated, and prioritized—all from a hypothetical attacker's point of view. The purpose of threat modeling is to provide defenders with a systematic analysis of the probable attacker's profile, the most likely attack vectors, and the assets most desired by an attacker." *

*Wikipedia contributors. "Threat model." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 22 May. 2019. Web. 19 Aug. 2019.

What is Threat Modeling? - 2

Asset - a resource of value, or something that an attacker wants to access, control, or destroy
Threat - a potential occurrence of an event or events that might damage or compromise an asset or objective
Vulnerability - a weakness in some aspect or feature of a system that makes an exploit possible
Attack - an action taken that utilizes one or more vulnerabilities to realize a threat to compromise or damage an asset

Defender View vs. Attacker View

- Which assets to protect?
- What vulnerabilities to fix?
- How deep should cyber defense be?

Source: Shostack, A. *Threat Modeling: Designing for Security.* Wiley, 2014. ISBN 978-1118809990.

Types of Attacks

Spoofing (client, process, data flow)

Tampering (process, data flow, data store)

Repudiation (process, data store)

Information Disclosure (excavation, interception, elicitation)

Denial of Service (data flow, data store)

Elevation of Privilege (privilege abuse, authentication abuse and bypass)

Social Engineering Attack (spear-phishing, exploiting trust)

Supply Chain Attack (modification during manufacturing, manipulation during distribution)

Hardware Integrity Attack (hacking, malicious update)

Injection Attack (resource, code, traffic, object)

Obstruction Attack (route disabling, orbital jamming, physical destruction of component)

Source: CAPEC, MITER
Source: Shostack, A. Threat Modeling: Designing for Security. Wiley, 2014. ISBN 978-1118809990.

Threat Modeling Methods

P.A.S.T.A

STRIDE Threat Types

Desired Property	Threat	Definition
Authentication	Spoofing	Impersonating something or someone else
Integrity	Tampering	Modifying code or data without authorization
Non-repudiation	Repudiation	The ability to claim to have not performed some action against an application
Confidentiality	Information Disclosure	The exposure of information to unauthorized users
Availability	Denial of Service	The ability to deny or degrade a service to legitimate users
Authorization	Elevation of Privilege	The ability of a user to elevate their privileges with an application without authorization

Microsoft STRIDE Threat Types

LINDDUN

Denning, Friedman, Kohno
The Security Cards:
Security Threat Brainstorming Toolkit

Jane Cleland-Huang's
Persona non Grata
http://www.infoq.com/articl
es/personae-non-gratae

Attack Tree

ACTA* Process as a Threat Modeling Framework?

Threat Modeling vs Threat Assessment: why we need both (1)

Similarities:

- Use the same sources of information:
  - Design and requirement documents
- Identify the system's critical assets, trust boundaries and vulnerabilities
- Use the similar approaches for a system analysis:
  - MBSE and Mission Threads
- Enumerate and prioritize threats to a system

*Adversary Cyber Threat Assessment (ACTA)

ACTA Process as a Threat Modeling Framework?

Threat Modeling vs Threat Assessment: why we need both (2)

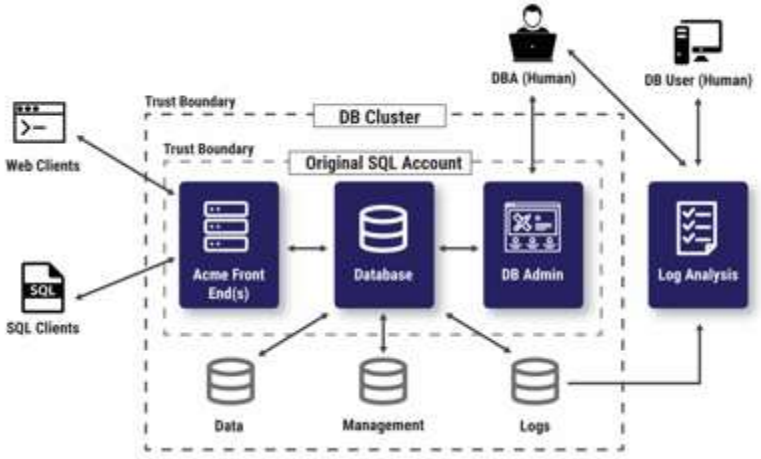
Differences:

- Threat modeling provides method(s) to identify threats
- Threat modeling methods help to identify how an attacker can accomplish his/her goal, which influence mitigation strategies
- Threat modeling requires more detailed view of a system in order to model a threat
- Threat assessment performs impact analysis of threats and vulnerabilities on a system and an organization
- Threat assessment identifies security and compliance requirements

Extend ACTA with STRIDE

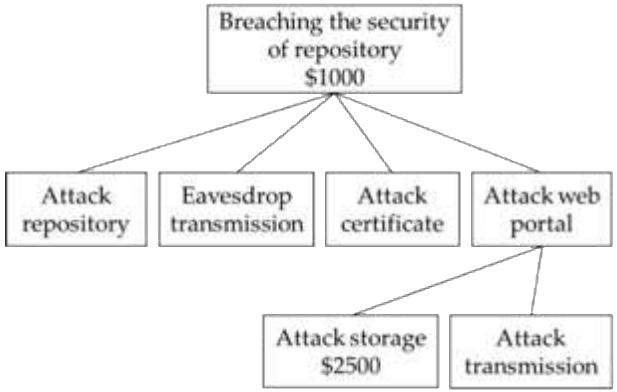
- Model the system: identify system entities, events, and boundaries of the system.
- Find threats: Answer the question *"what can go wrong with the system we're working on"*
- Variants are STRIDE per Element, STRIDE per Interaction

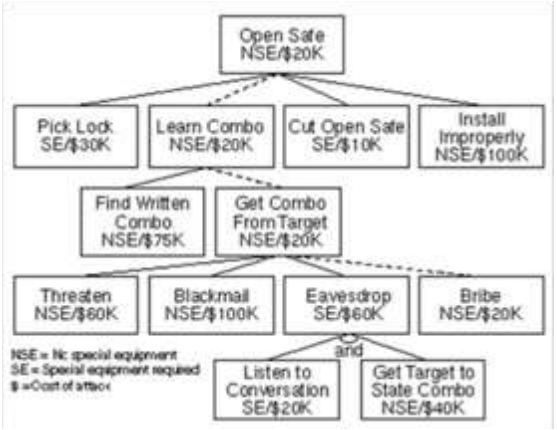
Threat	Property
Spoofing	Authenticity
Tampering	Integrity
Repudiation	Non-Repudiation
Information Disclosure	Confidentiality
Denial of Service	Availability
Elevation of Privilege	Authorization



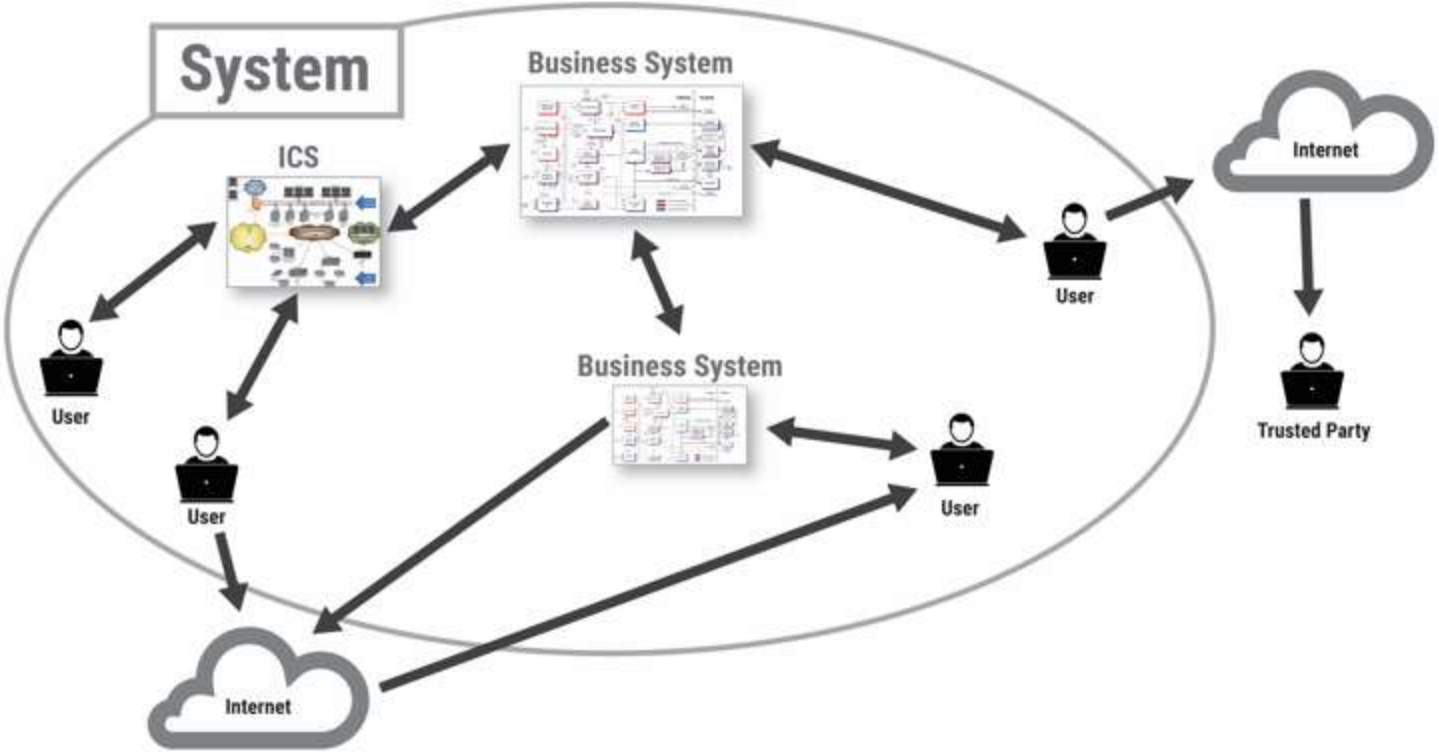
Extend ACTA with Attack Tree

- Diagrams attacks on a system in tree form.
- The tree root - the goal for the attack.
- The leaves - ways to achieve the goal.
- Each goal is represented as a separate tree.

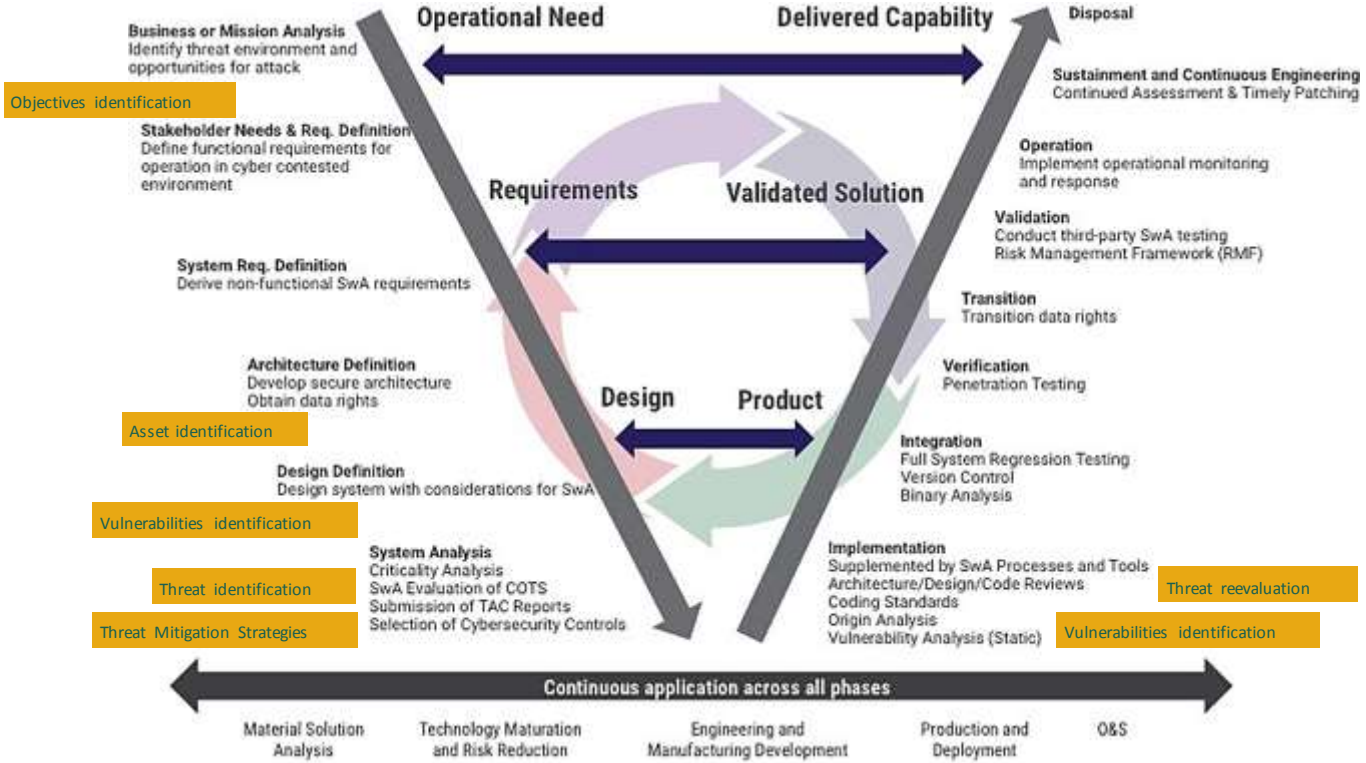




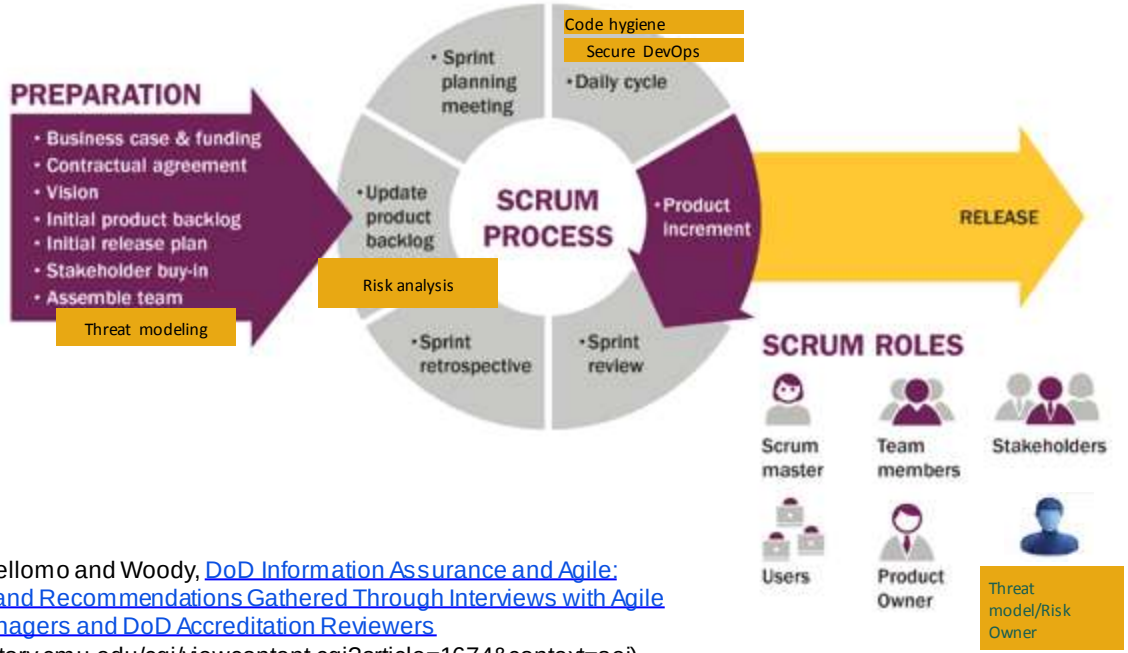
Threat Modeling for the System of Systems



Threat Modeling in Software Development Life Cycle



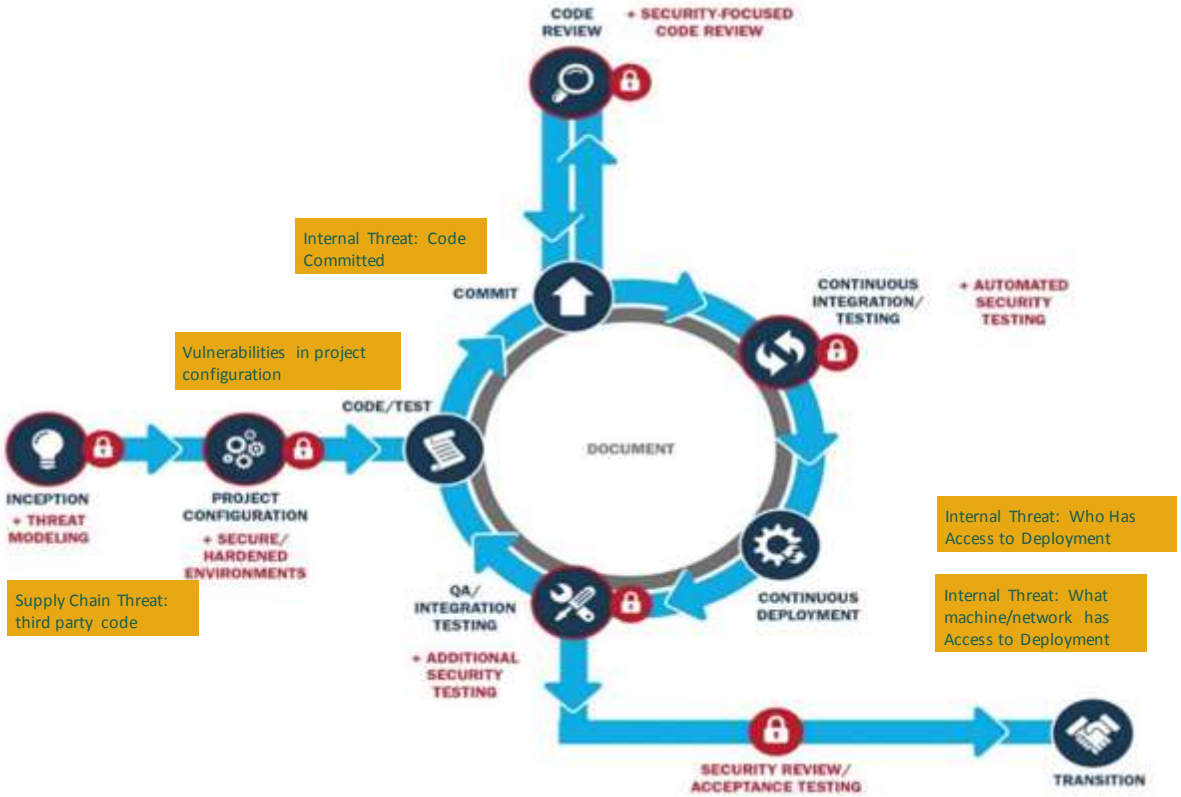
Threat Modeling in Agile



1. Code hygiene – introduce secure coding
2. Secure DevOps – include security tools
3. Threat modeling – represent a new role
4. Risk analysis – prioritize in backlog

(See also: Bellomo and Woody, DoD Information Assurance and Agile: Challenges and Recommendations Gathered Through Interviews with Agile Program Managers and DoD Accreditation Reviewers (http://repository.cmu.edu/cgi/viewcontent.cgi?article=1674&context=sei)

Threat Modeling in DevSecOps



Threat Modeling Method

Questions

Model-based Software Engineering for Cybersecurity: USAF High Assurance and DevSecOps

Dr. Carol Woody

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Several Major Air Force Program are Facing Similar Issues

- Hardware-based solution => Software-intensive system
- Waterfall methodology => Agile DevSecOps approach
- Program owned infrastructure => AF shared infrastructure (Platform One, Ski Camp)
- ATO for 3 years => Continuous ATO with integrated monitoring
- Static certified and unchanged within the Nuclear boundary => Software-based periodically refreshed Nuclear Surety environment

Today: Program Offices Whac-A-Mole

Winning in Features and Warfighter Effectiveness, but Losing in Defensibility and Stability

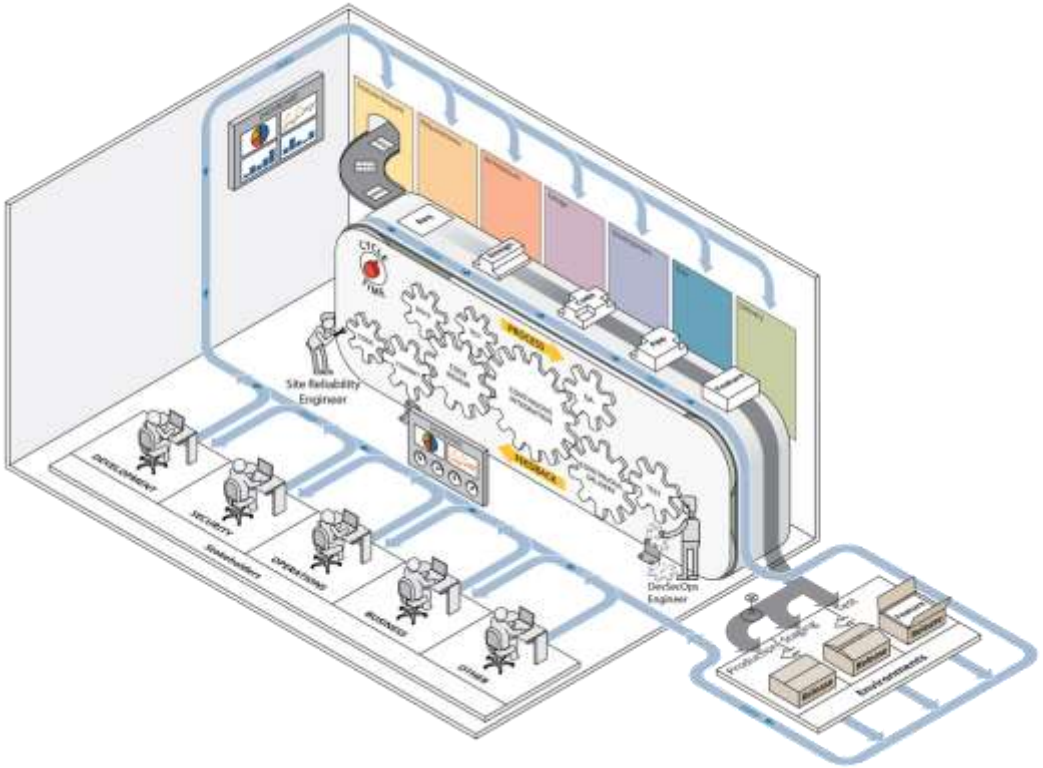
In June of 2020 a generally successful DoD program completed an 8 week “Hardening the Software Factory” effort in order to address accumulated technical debt and to address insufficient security and operations practices due to the narrow focus on speed of delivery.

These things occur, even in small relatively successful programs, when technical debt and insufficient security and operational practices are in place due to lack of knowledge, experience, and reference material to fully design and execute an integrated DSO strategy in which all stakeholder needs, including cybersecurity, are addressed.

Without the ability to perform formal analysis of a system’s numerous parameters, program offices are forced to play Whac-A-Mole and hope for the best.

What is DevSecOps?

A cultural and engineering practice that breaks down barriers and opens collaboration between development, security, and operations organizations using automation to focus on rapid, frequent delivery of secure infrastructure and software to production. It encompasses intake to release of software and manages those flows predictably, transparently, and with minimal human intervention/effort [1].

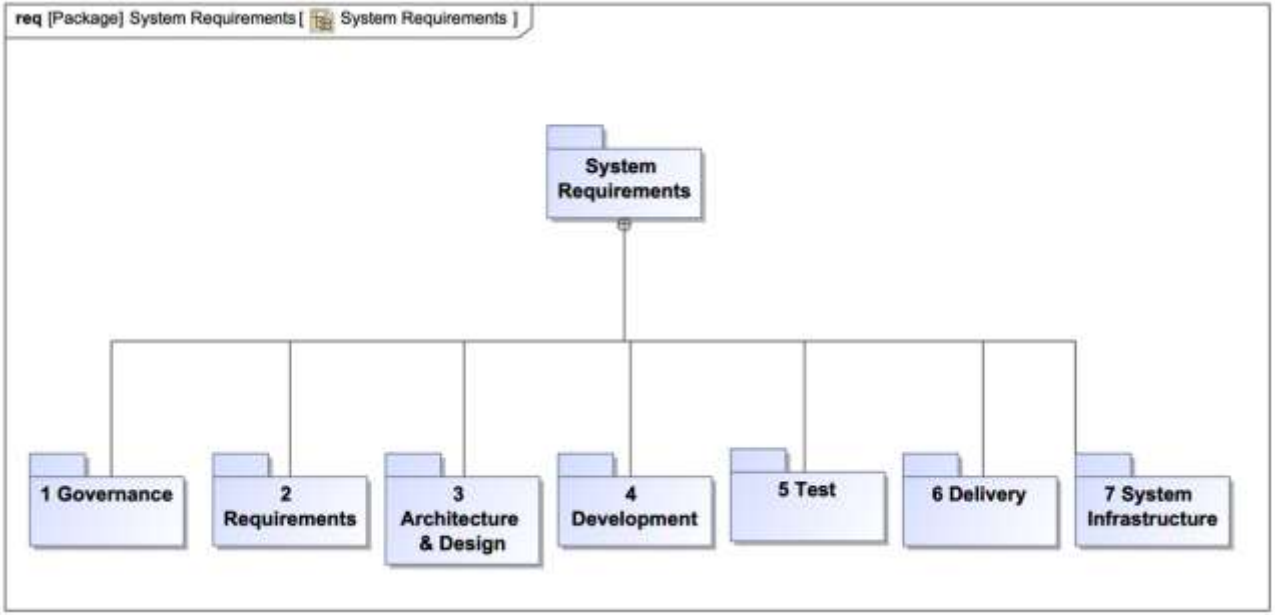


[1] DevSecOps Guide: Standard DevSecOps Platform Framework. U.S. General Services Administration. https://tech.gsa.gov/guides/dev_sec_ops_guide. Accessed 17 May 2021.

DevSecOps Maturity Levels

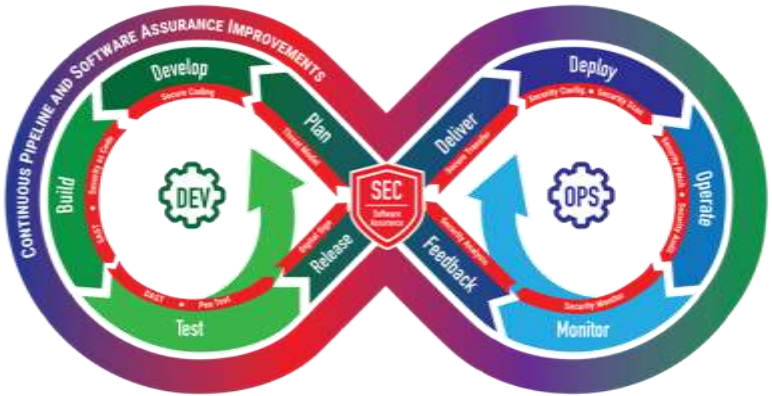
Term	Documentation
Maturity Level 1	Performed Basic Practices: This represents the minimum set of engineering, security, and operational practices that is required to begin supporting a product under development, even if only performed in an ad-hoc manner with minimal automation, documentation, or process maturity. This level is focused on minimal development, security, and operational hygiene.
Maturity Level 2	Documented/Automated Intermediate Practices: Practices are completed in addition to meeting the level 1 practices. This level represents the transition from manual, ad-hoc practices to the automated and consistent execution of defined processes. This set of practices represents the next evolution of the maturity of the product under development's pipeline by providing the capability needed to automate the practices that are most often executed or produce the most unpredictable results. These practices include defining processes that enable individuals to perform activities in a repeatable manner.
Maturity Level 3	Managed Pipeline Execution: Practices are completed in addition to meeting the level 1 and 2 practices. This level focuses on consistently meeting the information needs of all relevant stakeholders associated with the product under development so that they can make informed decisions as work items progress through a defined process.
Maturity Level 4	Proactive Reviewing and Optimizing DevSecOps: Practices are completed in addition to meeting the level 1-3 practices. This level is focused on reviewing the effectiveness of the system so that corrective actions are taken when necessary, as well as quantitively improving the system's performance as it relates to the consistent development and operation of the product under development.

DevSecOps Requirements Map to Maturity



Legend / Trace	Maturity Level 0	Maturity Level 1	Maturity Level 2	Maturity Level 3	Maturity Level 4
System Requirements		15	75	51	3
1 Governance		22	33	19	
Gov_1 Track Changes Associated to Requirements		1	↗		
Gov_2 Track Progress with Scrum/Kanban Boards			1	5	3
Gov_3 Task Creation			2	4	
Gov_4 Metrics				4	3
Gov_5 Knowledge Management			15	9	3
Gov_6 System Assurance			2	4	10
Gov_7 Defect and Issue Tracking				2	
Gov_8 Noncompliance Tracking				3	
Gov_9 Document and Manage Identified Risks			2	1	
2 Requirements			8	6	2
Req_1 Document Requirements			4	6	
Req_2 Requirements Abstraction Layers		1	↗		
Req_3 Requirements Prioritization		1	↗		
Req_4 Requirements Validation		1		↗	
Req_5 Change Management of Requirements			1		1
Req_6 Requirements Authorization		1	↗		
3 Architecture & Design			4	1	5
Arc_1 Requirement Mapping		1	↗		
Arc_2 Implementation Mapping		1		↗	
Arc_3 MBSE					2
Arc_4 System Assurance Design			3		3
4 Development			13	18	5
Dev_1 Mapping to Requirements		1	↗		
Dev_2 Mapping to Architecture		1	↗		
Dev_3 Mapping to Tests		1	↗		
Dev_4 Secure Software Development				7	
Dev_5 Code Reviews			1	1	
Dev_6 Orchestration				1	2
Dev_7 Configuration Management			8	9	2
Dev_8 Integrated Development Environment (IDE)		1	↗		
Dev_9 Development Information Radiator		1		↗	
5 Test			1	6	4
Tes_1 Manual Testing			4		
Tes_2 Requirement Association		1		↗	
Tes_3 Automated Testing			1	2	2
Tes_4 Code Coverage		1		↗	
Tes_5 Penetration and Fuzz Testing		1		↗	
Tes_6 Testing Information Radiator		1	↗		
Tes_7 Multi-phase Testing				2	
6 Delivery			2	1	4
Del_1 Release Management		1		↗	
Del_2 ITSM Service Desk		1		↗	
Del_3 Continuous Delivery					2
Del_4 Product Recovery		1	↗		
Del_5 System Recovery		1	↗		
Del_6 Configuration Item Integrity		1		↗	
7 System Infrastructure			1	10	12	1
Sys_1 System's Nonfunctional Requirements		1	↗		
Sys_2 Automated Provisioning		1		↗	
Sys_3 System Maintenance				2	1	2
Sys_4 Communication		1		↗	
Sys_5 Information Management				6	5	
Sys_6 Infrastructure Configuration Management				2	1	
Sys_7 Automated Patch Management					3	1

What is a DevSecOps Pipeline

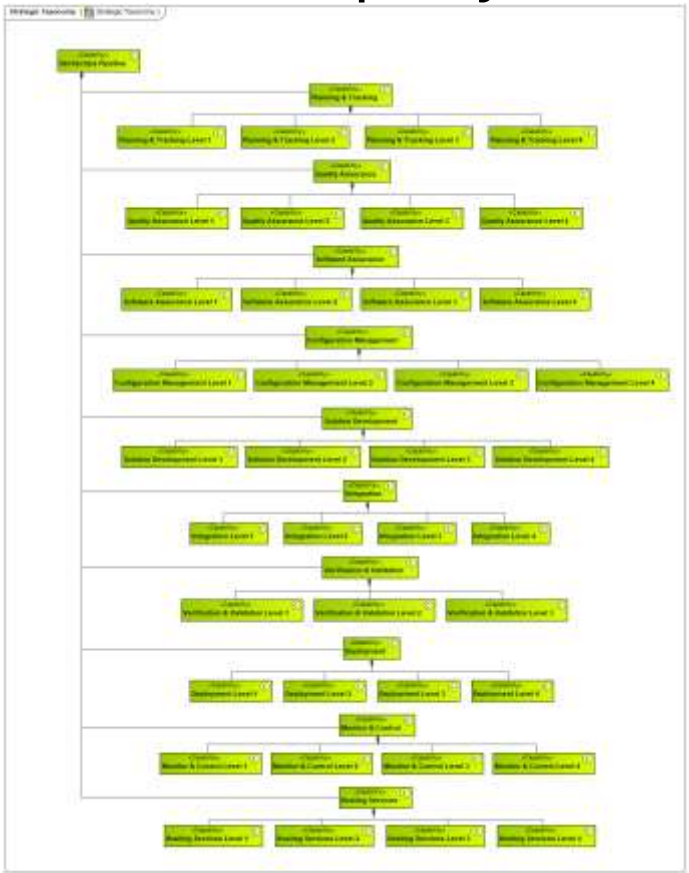


The DevSecOps pipeline is a socio-technical system composed of both software tools and processes. As the capability matures, it seamlessly integrates three traditional factions that sometimes have opposing interests:

- development; which values features
- security, which values defensibility
- operations, which values stability

A DevSecOps pipeline emerges when continuous integration of these three factions is used to meet organizational, project, and team objectives and commitments.

As a DevSecOps system matures, so will its capabilities

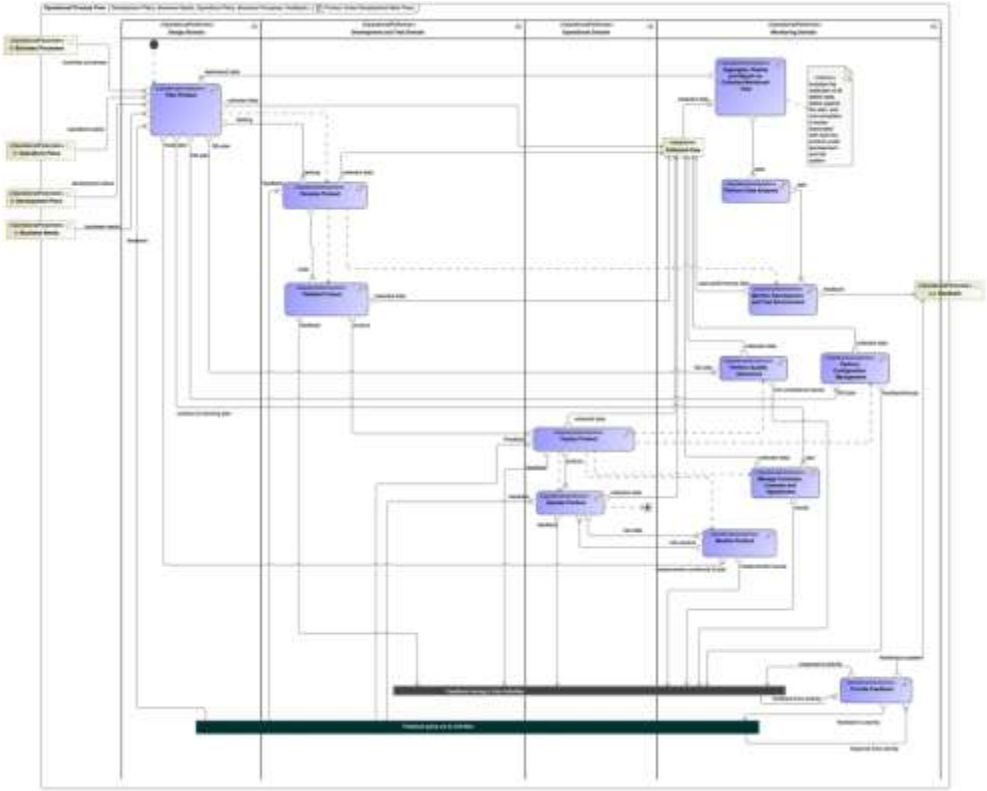


Legend

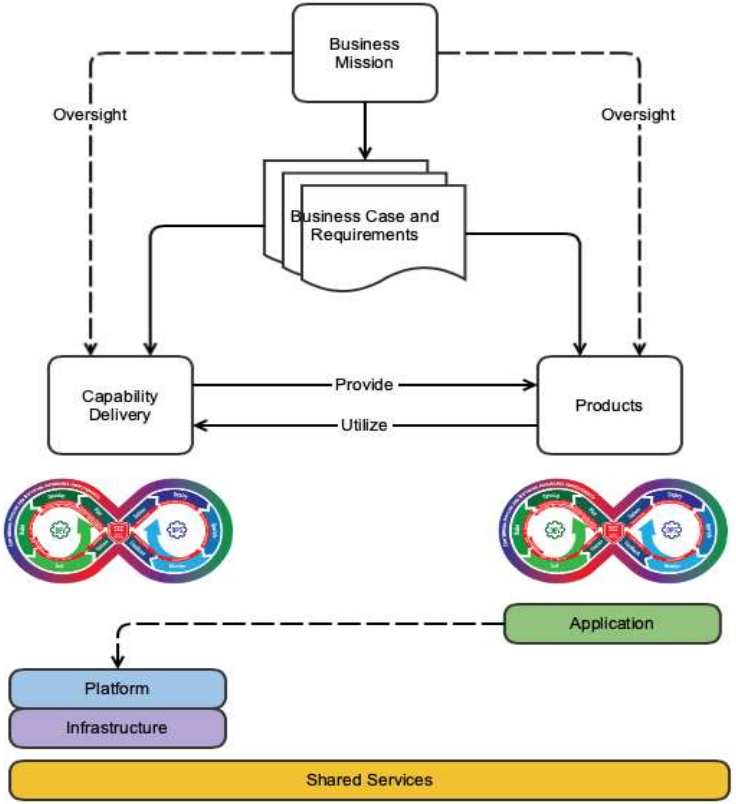
↗ Trace

Strategic Taxonomy	1 Governance	2 Requirements	3 Architecture & Design	4 Development	5 Test	6 Delivery	7 System Infrastructure
DevSecOps Pipeline							
Configuration Management	3	2		15	3	1	4
Deployment						6	1
Hosting Services	2	1			1		26
Integration				4			
Monitor & Control	27			3	3		12
Planning & Tracking	27	3				1	1
Quality Assurance	11			3	1		
Software Assurance	6	1	6	15	2	1	9
Solution Development	3	9	10	13			1
Verification & Validation	1	1		3	12		1

Cybersecurity Is an Acquisition Lifecycle Challenge and the Pipeline is only a Piece of the Puzzle



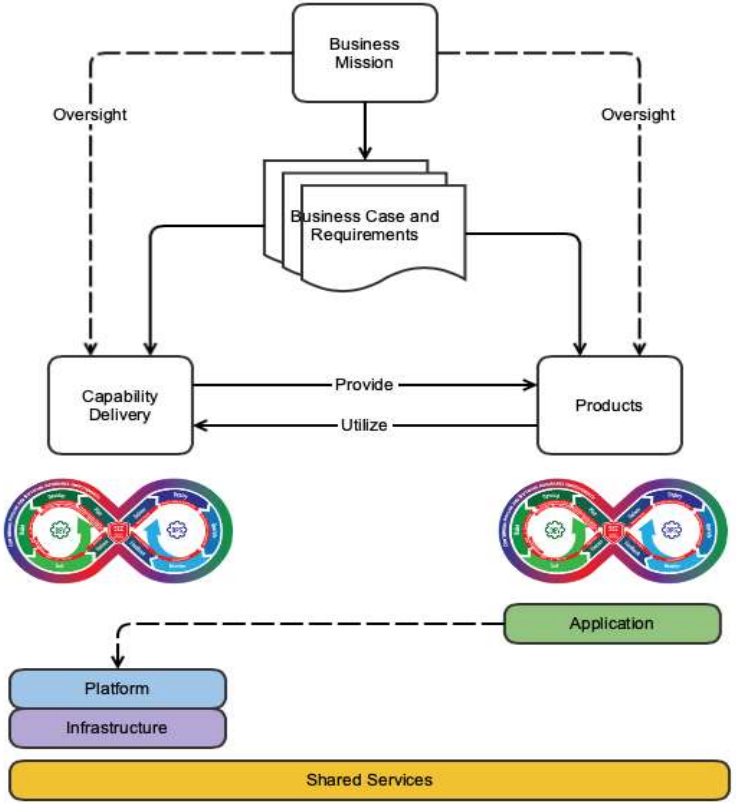
Challenge 1 for DSO: connecting process, practice, & tools



Creation of the DevSecOps (DSO) pipeline for building the product is not static.

- Tools for process automation must work together and connect to the planned infrastructure
- Everything is software and all pieces must be maintained but responsibility will be shared across multiple organizations (Cloud for infrastructure, 3rd parties for tools and services

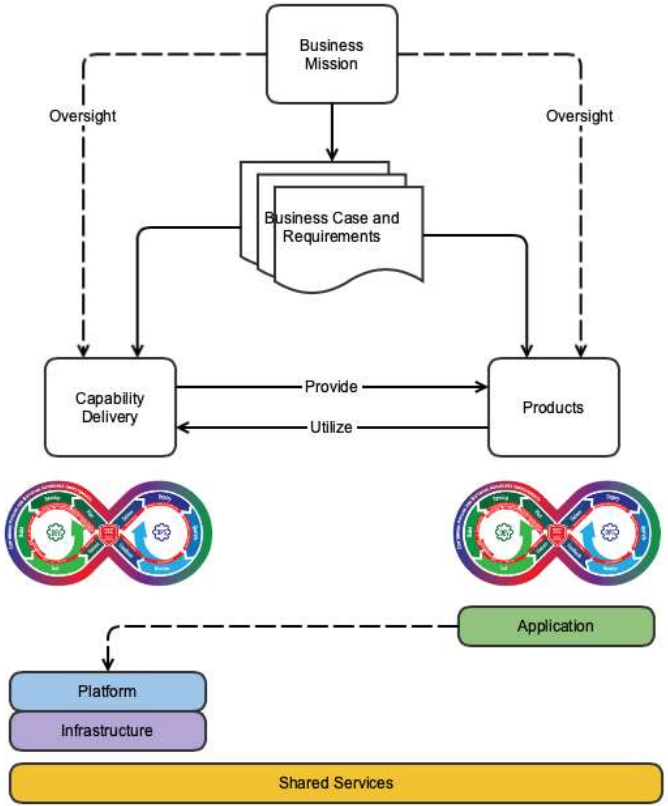
Challenge 2 for DSO: cybersecurity of pipeline and product



Managing and monitoring all of the various parts to ensure the product is built with sufficient cybersecurity and the pipeline is maintained to operate with sufficient cybersecurity is complex. Cybersecurity demands effective governance to address:

- What trust relations will be acceptable, and how will they be managed?
- What flow control and monitoring are in place to establish that the pipeline is working properly? Are these sufficient for the level of cybersecurity required?
- What compliance mandates are required? How are they addressed by the pipeline? Is this sufficient?

What Are We Trying to Do…?

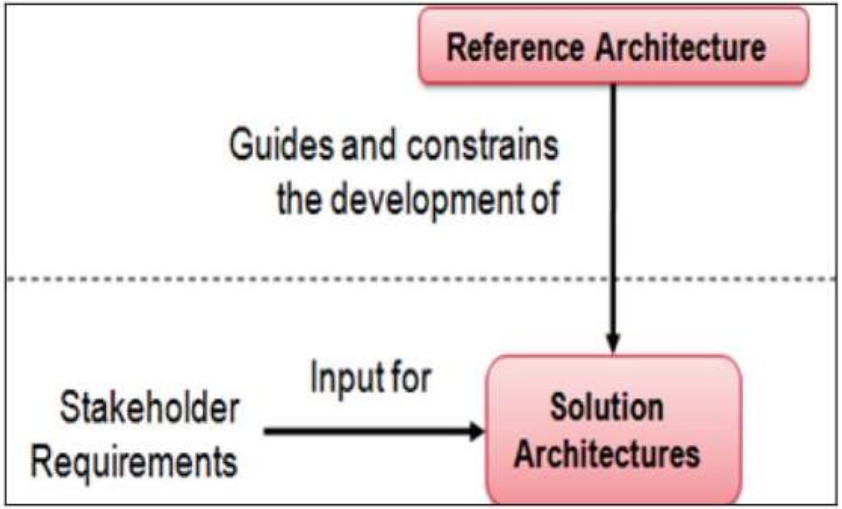


Create a Platform Independent Model (PIM) of a DevSecOps (DSO) System in order to be able to:

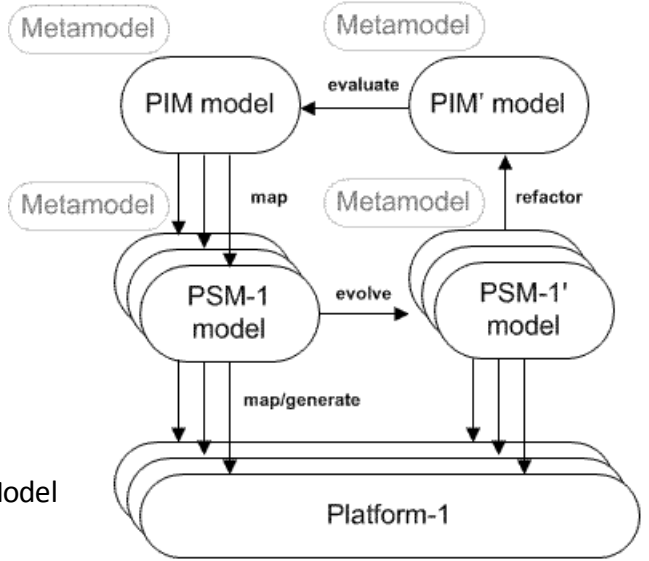
- Specify the DSO requirements to the lead system integrators who need to develop a platform-specific weapon solution that includes the weapon system and CI/CD pipeline
- Assess and analyze alternative pipeline functionality and feature changes as the weapon system evolves
- Apply DSO methods to complex weapon systems that do not follow well-established software architectural patterns commonly used in industry
- Provide a basis for threat and attack surface analysis to build a cyber assurance case

Reference Architecture/Platform Independent Model (PIM)

A **Reference Architecture** is an authoritative source of information about a specific subject area that guides and constrains the instantiations of multiple architectures and solutions [2].



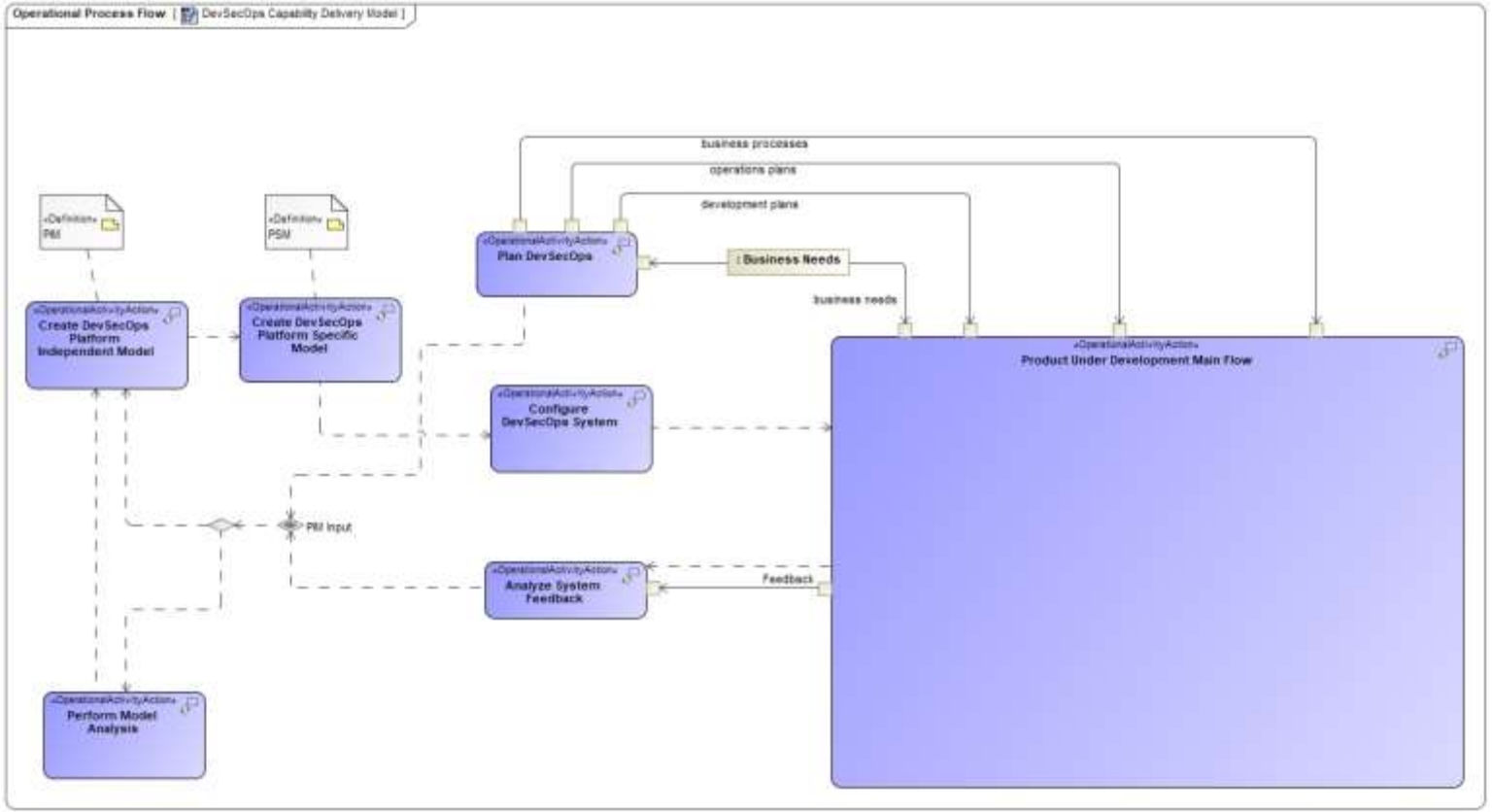
A PIM is a general and reusable model of a solution to a commonly occurring problem in software engineering within a given context, and is independent of the specific technological platform used to implement it.



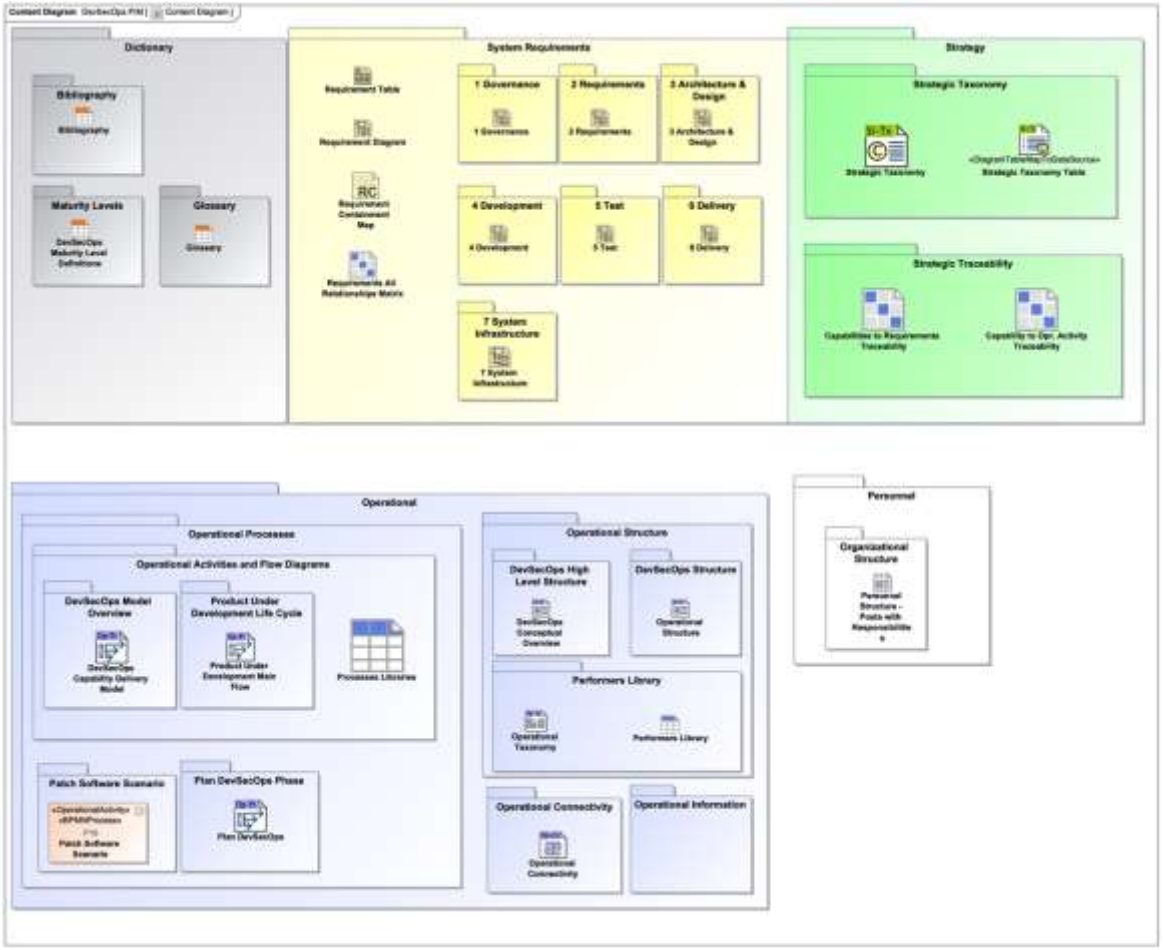
NOTE: PSM = Platform Specific Model

[2] DoD Reference Architecture Description, https://dodcio.defense.gov/Portals/0/Documents/DIEA/Ref_Archi_Description_Final_v1_18Jun10.pdf

Using the PIM

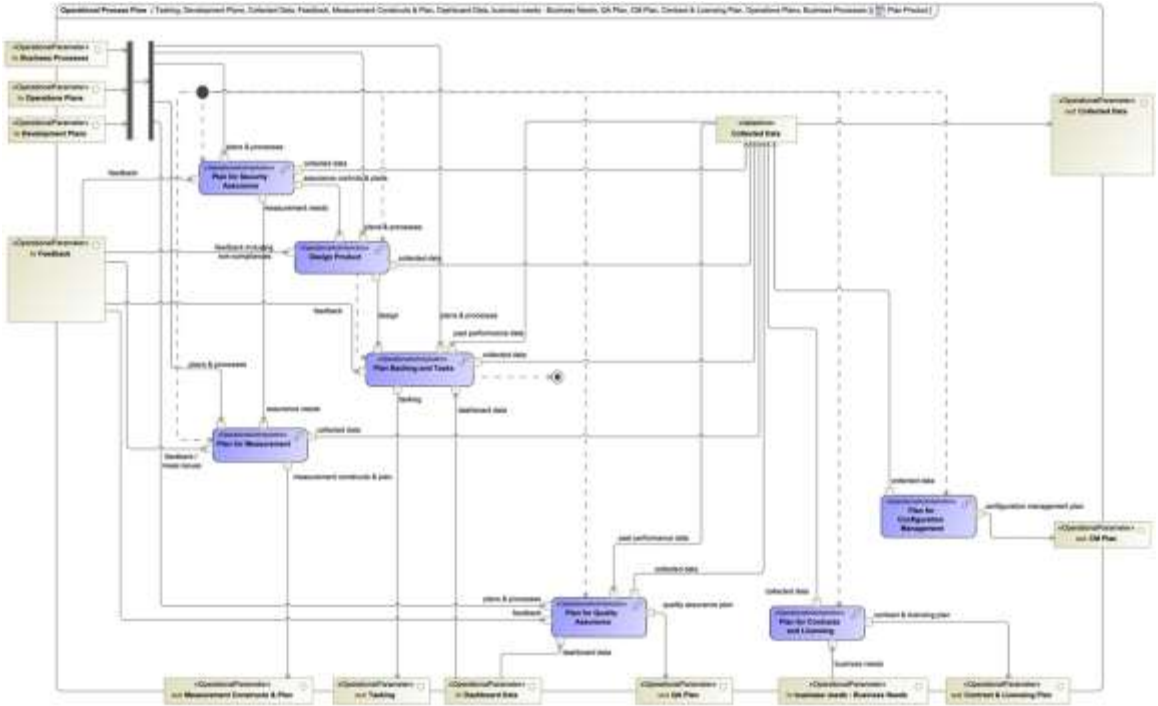


PIM Content



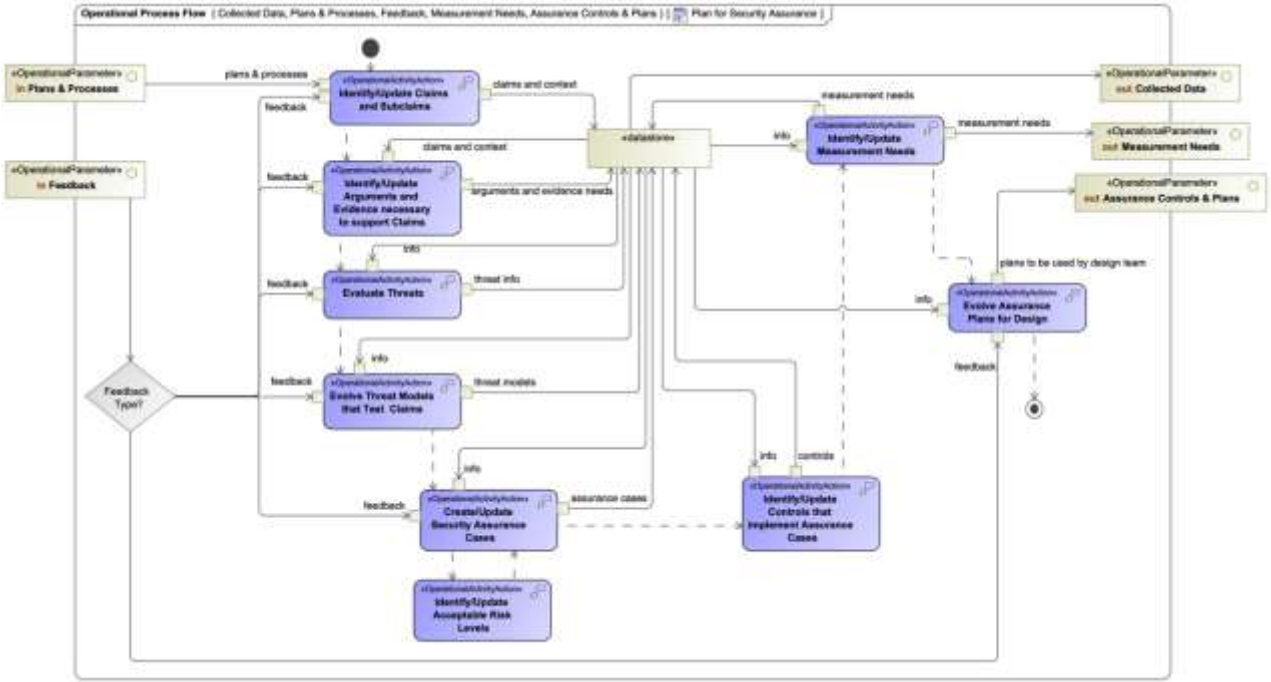
Building a Continuous Assurance Case

Continuous Planning



Building a Continuous Assurance Case

Assurance Case



Future: Program Office Topple

PIM will explicitly identify points (e.g. requirements, constraints, and conditions) that should be addressed or mitigated as well as mechanisms to manage coverage of the points. PSMs will present solutions for that. Using provided mechanisms will allow for the comparison of PSMs, analyzing of trade-offs and balancing the system dynamically.

Combining the DSO PSM with the system's architecture to build the single architecture, enables program offices to become organizations driven by smart automation, where delivery of a secure and resilient application quickly is the objective.

Through proper balance, programs will be able “to maintain a constant pace (i.e., play Topple) indefinitely.” [3]

[3] Principles behind the Agile Manifesto, https://agilemanifesto.org/principles.html

MBSE PIM

Questions

STPA-SafeSec and Assurance Cases

John Goodenough

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

STPA-SafeSec and Assurance Cases

STPA: An analysis method for identifying:

- Hazardous control actions leading to system losses
- Constraints needed to prevent hazardous control actions

Assurance Cases: A structured argument linking evidence to a claim about a system

- Explains why evidence is meaningful
- Helps in finding oversights and poor reasoning

STPA-SafeSec Scenarios and Assurance Cases

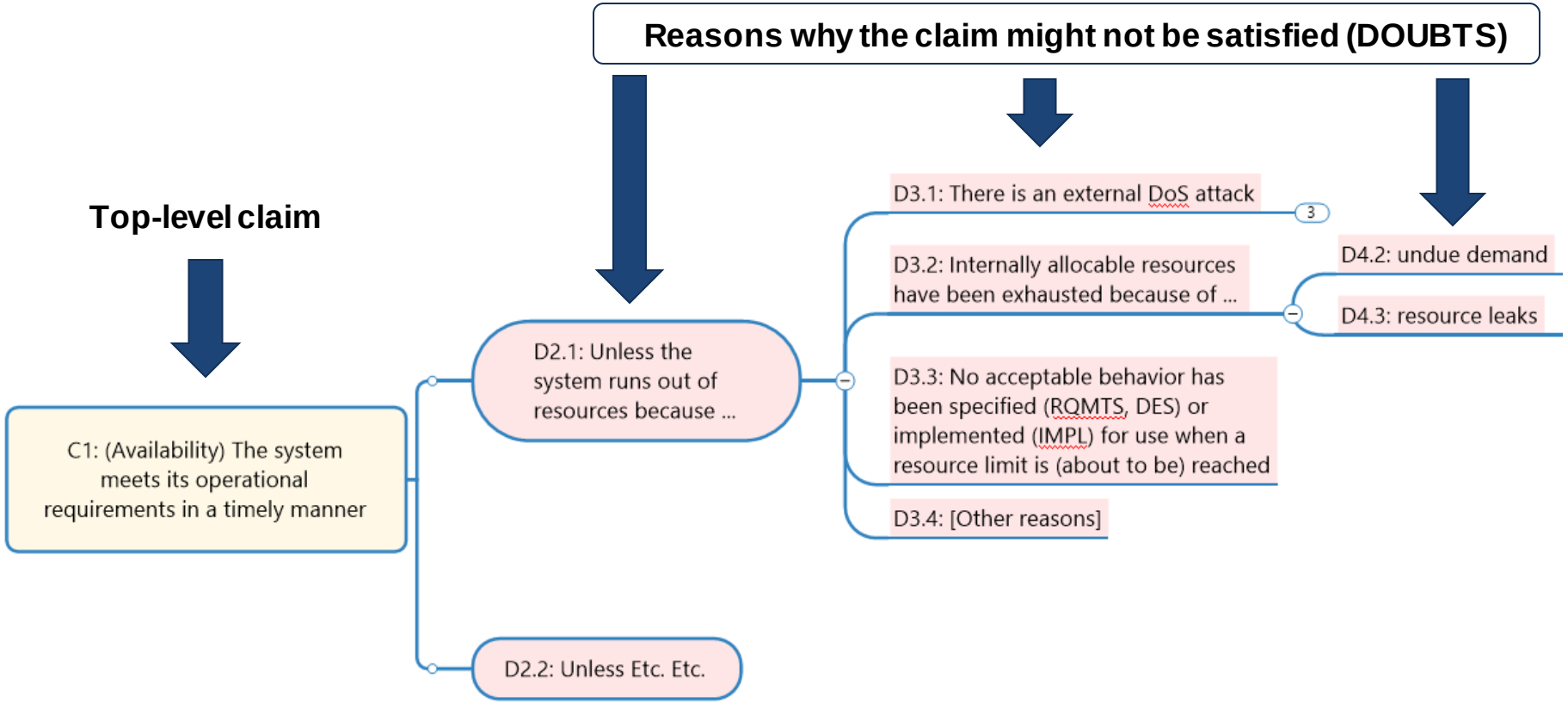
STPA scenarios provide a structured argument organizing analysis results

- "[It can be difficult] for external personnel to understand and use [STPA] analysis results" (p. 13, Friedberg, I. et al., STPA-SafeSec: Safety and security analysis for cyber-physical systems, J. of Info. Sec. and Applications (2016))

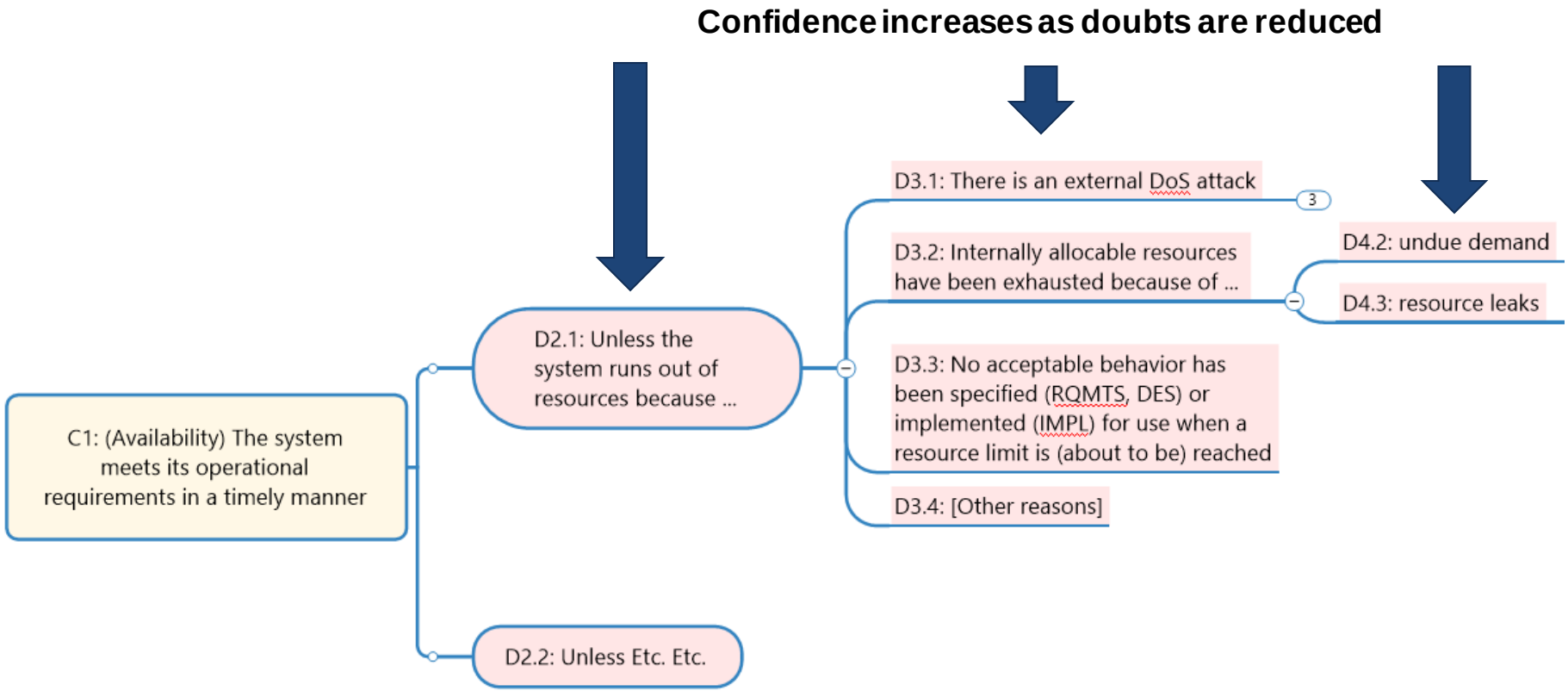
Assurance cases (in STPA context)

- Conceptually similar to STPA scenarios
- Gaining assurance, through evidence, that STPA constraints, as implemented, will prevent system losses
- Can complement STPA analyses

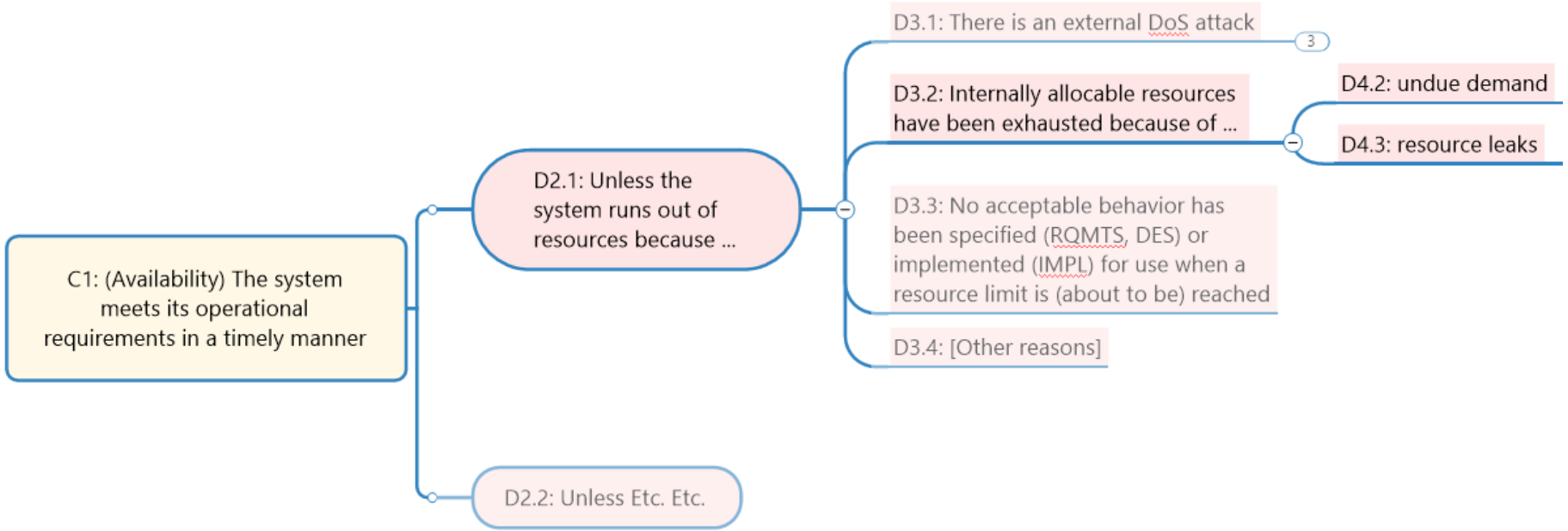
Example: Claim and Defeaters



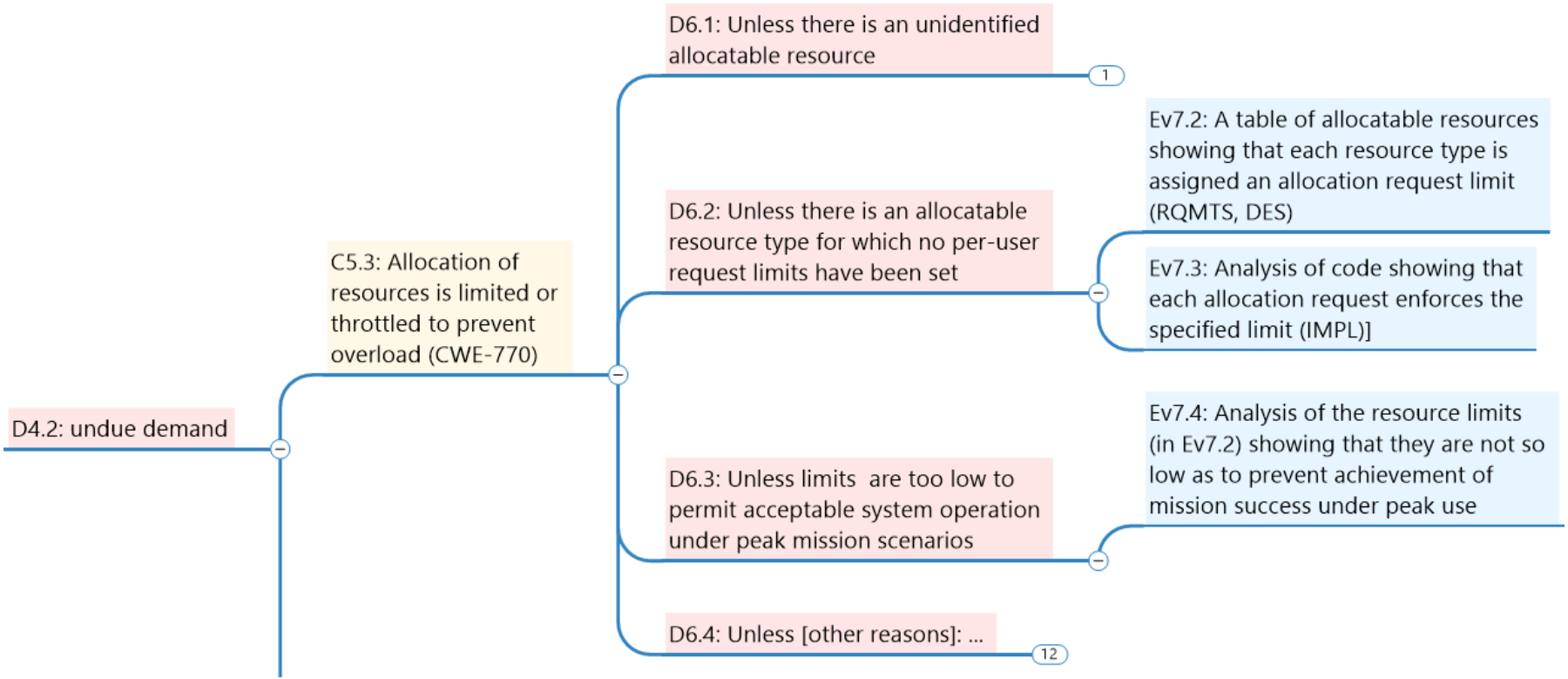
Example: Claim and Defeaters



Example: Claim and Defeaters



Reducing Doubt (with Evidence)



Summary

We are exploring:

- How the AC can identify exit criteria for a stage in the DevSecOps pipeline
- How to determine what evidence needs to be refreshed to maintain confidence that (relevant) exit criteria continue to be met after a change (the reassurance case)

Have had discussions with the System Software Security Engineering (S3E) Consortium about using this approach for the most egregious common weaknesses found in actual systems (top 25 CWEs)

Assurance Cases

Questions

CERT GBSD Projects Summary

Dr. Carol Woody

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Next Steps for CERT Tasks

Funded CERT FY22 Focus:

- Software assurance planning
- Security of the software supply chain
- DevSecOps software assurance

Opportunities for Cybersecurity Integration with Architecture

- Zero trust in design
- Threat modeling in architecture analysis
- Analyze GBSD program specific model against the Program Independent Model (PIM)
- Assurance Cases with STPA analysis

Contact Information

Carol Woody, Ph.D.

cwoody@cert.org

Web Resources
https://sei.cmu.edu/